uc3m | Universidad Carlos III de Madrid

Grado Universitario en Ingeniería Informática
2017-2018

Trabajo Fin de Grado

Análisis de logs del sistema para la realización de un estudio sobre seguridad y comportamiento

---

Daniel Cano Merchán

Tutor/es

Ana Isabel González-Tablas Ferreres

04-10-2018

Índice de contenidos

1. Contenido del documento

Índice de ilustraciones

Índice de tablas

Índice de figuras

Agradecimientos

A mis padres y Sheila que aguantaron la tormenta conmigo, a Mario amigo incansable, a esas cervezas con la gente que me acompañó y que no entendían lo que hacía.

“Y una vez que la tormenta termine, no recordarás como lo lograste, como sobreviviste. Ni siquiera estarás seguro si la tormenta ha terminado realmente. Aunque una cosa si es segura, cuando salgas de esa tormenta, no serás la misma persona que entró en ella. De eso se trata la tormenta.”

Haruki Murakami

Prólogo

Una necesidad básica para nosotros, es estar seguros.

Desde que somos pequeños buscamos estar protegidos en todos los aspectos de nuestra vida. Es innegable que hoy en día la tecnología y la informática forman una parte fundamental de nuestro día a día.

Desde hace años la sociedad ha cambiado a una sociedad más conectada, más tecnológica una sociedad en la que el camino al trabajo o a nuestra casa cruzamos multitudes de dispositivos.

Desde el coche, los semáforos, teléfonos, señales, el transporte público, bancos, tiendas…

Todos estos dispositivos se encuentran expuestos a numerosas amenazas con el fin de alterar su comportamiento u obtener datos acerca de nosotros y de lo que hacemos. Y a veces estas amenazas consiguen su propósito.

Para protegerse por tanto se requieren de metodologías, tácticas y estrategias que permitan proteger esta parte, que ya forma parte de nuestra vida, que es la tecnología.

El proyecto que se recoge en este documento trata sobre una auditoria de los logs recogidos de un servidor de la universidad Carlos III de Madrid, denominado Guernika.

Este proyecto pretender recoger parte de los problemas a los que se enfrenta un servidor real, expuesto a internet. El motivo de haber elegido este trabajo es aprender un poco más sobre ciberseguridad, ámbito profesional del que quiero aprender todo lo posible.

Desde que era programador antes de estudiar en la universidad siempre tuve curiosidad por la seguridad. Pero nada encendió la chispa hasta que un día en la empresa, recibimos una auditoría de seguridad.

Por supuesto el equipo al que pertenecía suspendió la auditoría de seguridad sobre el trabajo realizado y tuvimos que realizar una serie de parches para corregir las amenazas detectadas. Por suerte o por desgracia yo tuve que realizar tales parches de seguridad, al corregir tales amenazas y entender como estas amenazas afectaban a nuestro trabajo, me interesé por el mundo de la seguridad informática y de momento me ha llevado hasta aquí.

A volver a estudiar de nuevo y empezar una carrera con el fin de entender mejor la informática y a cambiar de trabajo donde ahora me dedico a la seguridad informática.

Glosario de términos

.gz y .tgz: Extensiones utilizadas para los ficheros *GZIP*, que se trata de un programa de software libre para comprimir archivos.

API: Interfaz de programación que ofrece una serie de funciones y servicios para ser utilizados en otro software.

Acceso físico: Referido al acceso que se tiene a una máquina cuando a este se puede acceder en persona, es decir se puede tocar y observar la máquina en concreto.

Acceso lógico: Referido al acceso que se realiza sin tener acceso físico es decir mediante programas o internet.

Amazon web services, AWS: Servicio cloud ofrecido por la compañía Amazon.

Apache Lucene: Modelo multiplataforma de búsqueda de texto de la fundación Apache.

Backup: Copia de seguridad.

Botnet: Conjunto de equipos comprometidos generalmente de forma ilegal para ser utilizados con un fin.

Cdrom: Disco compacto para almacenar información. Hoy en día en desuso.

Cloud: Tecnologías, métodos y servicios accesibles al usuario por internet.

CSV: Ficheros que contiene valores separados por un carácter, generalmente una coma.

CPU: Hardware de una computadora que se utiliza para interpretar y ejecutar las ordenes de un programa.

Cuaderno de bitácora: Cuaderno utilizado por los navegantes para anotar hechos relavantes en sus viajes

Debian: Sistema operativo de software libre basado en el kernel de Linux y las herramientas GNU.

Demonio: Serivicio que se ejecuta en segundo plano en los sistemas de tipo Unix.

Día cero o zero day: Ataque basado en el cocimiento de una vulnerabilidad desconocida.

Dirección IP: Identificativo de red de una computadora conectada que utilice el protocolo TCP/IP.

Entorno de escritorio: Se refiere al software y conjunto de herramientas utilizado para manejar una computadora a través de herramientas gráficas.

Firewall o cortafuegos: Software que limita el acceso de red a una computadora con el fin de protegerla.

Gdpr: General Data Protection Regulation, reglamento de protección de datos de ámbito europeo.

Gigabytes: Unidad que mide el almacenamiento de información equivalente a 10^9 bytes.

Git: Software utilizado para el control de versiones de un software.

Github: Entorno de desarrollo comunitario que sirve para alojar software basado en Git.

Gnome: Entorno de escritorio de software libre basado en las librerías GTK+.

Hardware: Componentes o materiales físicos que forman parte de una computadora.

Hadoop: Aplicación de la comunidad Apache utilizada en sistemas distribuidos.

HTTP: Protocolo de comunicación en red.

IOT: Internet of Things, se refiere al conjunto de dispositivos de baja capacidad computacional conectados a internet.

Kernel: Software crítico de un sistema operativo que recoge el conjunto de funcionalidades para manejar el hardware del equipo y ofrece a los programas una interfaz para manejarlos.

Linux: Kernel de software libre bajo GPL2 basado un Unix.

LOPD: Ley Orgánica de Protección de Datos cuyo objetivo es proteger los datos personales de los ciudadanos.

Machine learning: Campo de estudio de la inteligencia artificial cuya finalidad es que una máquina o computadora sea capaz de aprender.

Nmap: herramienta para realizar un escáner de puertos en una comutadora.

On-Premise y SaaS: SaaS se refiere a Software as a Service, en el que un usuario utiliza un servicio alojado en la nube generalmente por medio de credenciales o licencias. On Premise software que instala o ejecuta en el cliente.

Open source: Se refiere al código fuente o software que se puede modificar, estudiar y mejorar de manera legal.

Path: Término utilizado para referirse a la ruta donde se encuentra un ejecutable o fichero.

Plugin: Complemento software que añade funcionalidades nuevas al ya existente.

Puerto: Interfaz de red por la que se comunican los programas o sistemas.

Q1: Primer cuarto de un año fiscal.

Restful JSON: Modelo de software basado en la comunición software basado en HTPP y JSON.

Root: usuario administador en sistemas basados en Unix.

Sandboxing: Técnica de aislamiento de procesos o software en el que si una parte es comprometida no afecta al resto de procesos.

Script: Fichero que contiene instrucciones para ejecutarse con un fin determinado por su programación.

Servidor: Computadora que atiende peticiones de usuarios generalmente a través de la red.

Shell: Interprete de comandos.

Software: Conjunto de herramientas lógicas y programas que se utilizan en un equipo.

Super usuario: usuario que es capaz de operar sin restricciones en el sistema.

SQL: Structured Query Language, lenguaje de consultas programable.

Unix: Sistema operativo multiusuario de los años 70 base conceptual de algunos sistemas modernos.

YAML: Tipo de estructura de ficheros cuyo énfasis es la lectura fácil de la información por personas.

Windows: Sistema operativo para PC.

Windows azure: Servicio cloud ofrecido por la compañía Microsoft.

WPA2: protocolo de comunicación de red wifi.

X86: Arquitectura de computadores compatible con las instrucciones 8086.

Capítulo 1: Introducción al documento

"There are two types of companies: those that have been hacked, and those who don't know they have been hacked."

John T. Chambers, CEO of Cisco Systems

1.1. Contexto económico y social

Hoy en día uno de los aspectos de la informática que más impacto está teniendo en los últimos años es la seguridad informática. A menudo en los medios de comunicación aparecen noticias relativas a incidentes de seguridad, que además afectan a las personas de primera mano. Afectan a sus bancos, a sus coches, a sus teléfonos, sus hospitales y redes sociales. Estos incidentes afectan a sus vidas personales y muchos de estos incidentes no aparecen en ningún medio de comunicación, se trata de incidentes ocultos a la mayoría, que no tiene consciencia de ellos.

Los incidentes de seguridad han aumentado durante estos años [1] [2], debido a una multitud de factores:

- Popularidad de internet.

- Aumento del número de la cantidad de dispositivos conectados a internet.

- Multitud de negocios han cambiado su forma de ejercer su negocio moviendo parte o su negocio completo a soluciones cloud.

- Aumento de los incidentes de seguridad informática en un 32% en el Q1 de 2018 respecto a 2017 [1].

- Incrementos de las vulnerabilidades y *días zero* [3] [4].

- Aumento de los ciberataques a España. [5]

Este impacto no sólo está teniendo repercusión en el mundo técnico de la informática si no que cada vez afecta a un mayor número de personas y empresas, desde bancos, sistemas automáticos, coches inteligentes, smartphones, dispositivos iot entre otros.

En los últimos años se ha podido comprobar incidentes de seguridad graves y mediáticos como:

- Vulnerabilidades en procesadores x86 (2018): Meltdown, spectre. Netspectre. [6]
- Uso intensivo de Ramsonware 2017, Wannacry, Petya. [7]
- Minado de criptomonedas remoto. [8]
- Botnet IoT, mirai 2017 [9]
- Ataque contra Sony Pictures. [10]
- Brecha de seguridad en Yahoo 2017 [11]
- BlueBorne 2017 [12]
- Krack wifi WPA2 [13]

Ante todas estas amenazas uno de los métodos utilizados para defenderse y tener consciencia de los ataques es el ***logging***.

¿En qué consiste la técnica de Logging?

Consiste en almacenar una serie de sucesos que ocurren en una aplicación o sistema informático en un fichero denominado *log*.

¿Qué es un log y qué contiene?

El término Log, se trata de un anglicismo utilizado en el campo de la informática.

Cuyo origen se remonta al contexto militar naval inglés [14]. El término *Log* era utilizado para referirse al cuaderno de bitácora que utilizaba un capitán de navío para guardar un informe sobre los sucesos ocurridos en sus viajes, también se refería al uso del instrumento formado por una cuerda y una corredera(*Log*) para medir la velocidad en nudos de las embarcaciones. Era también utilizado como término para referirse al cuaderno (*Log book*) que se encontraba en los cuarteles militares donde se recogía la entrada y salida del personal militar.

Un Log en el campo de la informática, se trata de un fichero que contiene una cantidad indeterminada de sucesos que han ocurrido anteriormente en una aplicación o sistema informático.

Por lo tanto de existir una medida sobre la información que se guarda en un fichero esta serían los eventos.

Un evento es algo que ha ocurrido en el sistema y por un criterio que se definió previamente en la aplicación o sistema informático es considerado relevante y por tanto almacenado en el fichero *log*.

Problema general

El problema de almacenar los eventos que ocurren en una aplicación o sistema informático es la dificultad de analizar la cantidad de información que es almacenada en estos ficheros, principalmente por dos motivos:

1) Dimensiones y cantidad: El tamaño puede ser inmanejable para un ser humano, debido al gran tamaño que pueden alcanzar algunos logs ocupando varios Gigabytes de texto plano. La cantidad de logs puede ser muy alta llegando a tener multitud de logs de gran tamaño.

2) Formato: cada log puede ser único, por lo que la estructura del log no está estandarizada.

Estos dos puntos principalmente hacen que el análisis de logs se realice únicamente cuando se tiene la creencia de que algo ha ocurrido y se busca saber por qué ha ocurrido. Por lo que para abordar el problema de forma eficiente se deben de utilizar técnicas distintas a las que se han utilizado tradicionalmente en los últimos años.

1.2. Motivación

El contexto en el que se plantea este estudio está ubicado en Leganés, Madrid, en la Universidad Carlos III, en concreto en las instalaciones del laboratorio de Departamento de Informática donde se encuentra el servidor de aplicaciones Guernika.

Se trata de un servidor de aplicaciones que es utilizado por los alumnos y profesores de la universidad para realizar prácticas tanto dentro de las clases presenciales de la universidad como desde fuera de la universidad, por lo que este servidor es accesible mediante internet. El objetivo de tener acceso a internet en este servidor, es que los alumnos puedan probar sus prácticas o realizar ejercicios sin tener que acudir a la universidad, permitiendo así evitar desplazamientos y que el alumno tenga más flexibilidad a la hora de realizar sus tareas.

Además este servidor se utiliza dentro de la universidad como máquina estándar para realizar correcciones de prácticas de distintas asignaturas.

Las ventajas de utilizar Guernika es que elimina restricciones económicas pues el acceso es gratuito y el criterio de corrección es el mismo para todos los alumnos al utilizar un solo entorno de corrección.

La finalidad del estudio es realizar un análisis acerca de la colección de Logs que recoge Guernika de forma automática sobre su funcionamiento.

No se dispone de una meta en concreto si no que se ha dejado a elección del alumno encontrar un objetivo, por lo que el objetivo no está cerrado y forma parte de la tarea del alumno acotar y encontrar una meta final.

Para ello el alumno ha recibido una colección de Logs recogidos durante el curso.

1.3. Objetivos

El objetivo que se plantea en este documento es utilizar la colección de Logs recibida por parte del alumno para obtener conclusiones acerca de lo que ha ocurrido en Guernika a través de los Logs.

Siendo por tanto sus objetivos principales:

1. Encontrar la forma de manejar y tratar la información que se encuentra en los logs.
2. Utilizar la información contenida en los logs para realizar un estudio sobre lo ocurrido en Guernika.
3. Utilizar el estudio para realizar un informe sobre el estado de la seguridad en Guernika.

1.4.Marco regulador

Por defecto la mayoría de los servidores están configurados para almacenar en sus logs información acerca de los usuarios.

Una parte importante de esta información almacenada en los logs se trata de información personal. Por lo que este tipo de recolección de información está amparada bajo la normativa europea GDPR, aprobada el 14 de abril de 2016 y con vigencia desde el 25 de mayo de 2018.

Esta normativa reemplaza a la normativa anterior europea 95/46/EC. Por lo que todos los ciudadanos europeos están amparados bajo esta normativa.

Debido a que los logs utilizados son posteriores al 25 de mayo de 2018 la información que se encuentra almacenada en estos logs era correspondiente a la normativa anterior y por tanto el estudio realizado no confronta contra la normativa GDPR europea.

El servidor Guernika pertenece al conjunto de servicios propios de la Universidad Carlos III de Madrid, al utilizar los servicios propios de la Universidad Carlos III de Madrid se está aceptando las condiciones de uso recogidas en la normativa de la universidad.

Reglamento de Organización y Funcionamiento del Servicio de Informática, aprobado por la Junta de Gobierno en sesión de 17 de junio de 1997. [15]

Normativa de utilización de Aulas informáticas para actividades relacionadas con las enseñanzas de 3er ciclo. Aprobadas por la Comisión Gestora el 20 de junio de 1994. [16]

Normativa de utilización de las aulas informáticas por los alumnos de primer y segundo ciclo de enseñanzas universitarias. (Resolución 7/93 de 3 de marzo) [17]

Además debido a que la universidad Carlos III se encuentra bajo territorio Español, se encuentra sujeta también a la Ley Orgánica de Protección de Datos 15/1999 (LOPD) que fue revisada por el Real Decreto 1720/2007 que garantiza los principios de seguridad a aplicar en los sistemas de información para los datos de carácter personal de los ciudadanos.

Debido a que los datos están catalogados en tres tipos: Básico, Medio y Alto se procede a incluir un pequeño resumen en el que se describen estos tipos junto con una justificación al tipo al que pertenecen.

Tipo básico:

Persona física	Datos capaces de identificar a una persona física
Empleo	Datos que permitan obtener información acerca del empleo o tareas que realiza una persona

Tabla 1- LOPD datos de tipo básico

Tipo medio:

Ocupan los anteriores más los siguientes:

Registro penal	Datos capaces de identificar infracciones o sanciones penales.
Solvencia y crédito	Datos que permitan identificar el patrimonio y crédito de una persona.
Registro civil y seguridad social	Datos que contienen información acerca del registro civil y la seguridad social.
Características personales	Datos que permiten identificar rasgos personales de una persona o comportamientos propios de esta.

Tabla 2- LOPD datos de tipo medio

Tipo alto:

Ocupan todos los anteriores más los siguientes

Ideología	Datos capaces de identificar la afinidad ideológica de una persona, ya sean convicciones políticas, sexuales, religiosas o sindicales.
Fines policiales	Datos que permitan identificar el patrimonio y crédito de una persona.
Violencia de género	Datos que contienen información acerca de hechos involucrados en casos de violencia de género.

Tabla 3- LOPD datos de tipo alto

Por lo que los datos que aparecen en los logs se ubican en un nivel **BÁSICO**, que se corresponden con las siguientes medidas de seguridad que cumple la universidad Carlos III:

- Control de acceso lógico a los datos.
- Soporte y protección a los datos de personales.
- Notificación de incidencias.
- Backups.

Conclusiones acerca del marco regulador

No se encuentran impedimentos para el desarrollo de la actividad propuesta. Salvo que se publiquen los datos con el nombre de los usuarios reales, se debe de utilizar usuarios anónimos con el fin de utilizar los datos con objetivos científicos y estadísticos.

1.5.Estructura del documento

El documento está estructurado en 9 partes diferenciadas:

1. **Introducción**: Contiene la motivación del documento y el contexto tanto económico como legal en el que se ampara.
2. **Estado de la cuestión**: Se trata del primer esbozo que se realiza al abordar el problema se analiza el punto de partida y se concreta hacía donde se quiere llegar.
3. **Requisitos**: Análisis formal del problema.
4. **Diseño y configuración de herramientas**: Describe de forma detallada el diseño de la arquitectura utilizada así como de sus herramientas.
5. **Análisis de la información**: Trata sobre el estudio que se ha realizado sobre la información de los logs.
6. **Informe final**: En él se realiza el informe y consideraciones de seguridad final sobre Guernika.
7. **Gestión de proyecto**: Se recoge la planificación y los costes asociados al proyecto.
8. **Conclusiones y líneas futuras:** Consideraciones finales sobre el estudio realizado y mejoras futuras.
9. **English competitions.**

Capítulo 2: Estado del arte

El siguiente punto trata acerca el contexto del que parte el proyecto para justificar el estudio de las distintas herramientas que existen en el mercado para el análisis de logs. Se incluirá una descripción breve por cada herramienta junto con un listado de precios si procede.

2.1. Contexto inicial

Como punto de partida se dispone de una colección de Logs. Por lo que el primer paso para abordar el problema es entender que se ha recibido.

Información recibida

La colección que se ha recibido se ha recibido mediante un fichero comprimido en formato .tgz

Que contiene lo siguiente:

Una carpeta denominada *var*, que contiene la subcarpeta *log*, a su vez las subcarpetas:

- *apt*: contiene 13 logs tipo *history.log* y 13 logs tipo *term.log.* 12 de cada tipo comprimidos en formato .gz
- *firebird*: carpeta vacía.
- *fsck*: contiene dos ficheros checkfs y checkroot.
- *gdm3*: carpeta vacía.
- *installer*: Contiene la subcarpeta cdebconf con los ficheros questions.conf y templates.dat y en la raíz de installer hardware-summary, lsb-release, partman, status y syslog
- *lightdm*: lightdm.log y lightdm.log.old así como los ficheros x-0.log,x-0.log.old y x-0-greeter.log y x-0-greeter.log.old
- *munin*: 9 Logs bajo el nombre munin-node.log 7 de ellos comprimidos en formato .gz
- *ntpstats*: carpeta vacía.
- *Samba*: carpeta vacía.
- *speech-dispacher*: carpeta vacía.

Y en la raíz de la propia de la carpeta *log:*

1. 13 logs bajo el nombre de *alternatives.log* , 12 de ellos comprimidos en formato .gz
2. 356 logs bajo el nombre de *auth.log*, 355 comprimidos en formato .gz
3. 356 logs bajo el nombre de *daemon.log*, 355 comprimidos en formato .gz
4. 6 logs bajo el nombre de *debug.log*, 4 comprimidos en formato .gz
5. 14 logs bajo el nombre de *dpkg.log*, 12 comprimidos en formato .gz
6. 198 logs bajo el nombre de *kern.log*, 197 comprimidos en formato .gz
7. 224 logs bajo el nombre de *mail.log* 222 comprimidos en formato .gz
8. 6 logs bajo el nombre de *messages.log*, 4 comprimidos en formato .gz

9. 53 logs bajo el nombre de *syslog.log*, 52 comprimidos en formato .gz
10. 5 logs bajo el nombre de *user.log*, 4 comprimidos en formato .gz
11. 13 logs bajo el nombre de *wtmp.log*, 12 comprimidos en formato .gz
12. 4 logs bajo el nombre de *xorg.log*, 2 comprimidos en formato .gz

2.2. Análisis preliminar

Se ha realizado el siguiente análisis a priori de la información recibida en la colección de logs.

En primer lugar que se trata de un servidor que ejecuta un sistema operativo basado en Unix, debido a la estructura típica que mantiene de forma tradicional para almacenar los logs del equipo en el directorio */var/log.* Se observa por tanto que lo que se ha recibido es una copia del directorio */var/log* y que no se ha seleccionado de ninguna forma que información debe de analizarse.

Clasificación del sistema operativo y hardware

La carpeta */var/log/installer* contiene información acerca del momento en el que el sistema operativo fue instalado. Se desglosa por ficheros la identificación del sistema operativo y sus componentes:

/var/log/installer/lsb-realease:

Contiene información sobre la instalación del sistema operativo, un estudio del fichero revela que se trata de un sistema operativo GNU/Linux, en concreto, la instalación original fue un Debian con la versión del kernel 3.16.0.4 para procesadores x86_64, codename Jessie.

La instalación fue realizada el día 22-04-2015 mediante CdRom por lo que se trata de un servidor que probablemente se haya ido actualizando de versión a medida que han transcurrido los años pues a día de hoy llevaría más de tres años ejecutando Debian.

/var/log/installer/hardware-summary:

Contiene información acerca del hardware sobre el que se ejecuta el sistema operativo. El log revela que se trata del siguiente hardware:

- CPU: Intel Core i5-4460 3.20Ghz con 4 cores.
- Memoria RAM: 16367568 Kb que al convertirlos a GB son 16GB.

/var/log/installer/partman:

Ocupa la información acerca del particionado que se ha realizado durante la instalación de Debian. Contiene información útil del sistema pues indica cual es la estructura de los discos del equipo así como sus particiones.

El estudio del log revela que se trata de un equipo de un solo disco. */dev/sda* Que contiene las siguientes particiones:

- */dev/sda1* – sin particionar
- */dev/sda2* – C: Windows
- */dev/sda3* – D: Windows
- */dev/sda4* – Swap linux
- */dev/sda5* – Debian

Por lo que se deduce que el sistema operativo a estudiar mediante sus logs convive con otros sistemas operativos Windows.

Identificación del contenido de las carpetas

En este apartado se describe de forma breve el contenido de las subcarpeta carpetas que se encuentran en */var/log.*

- */var/log/apt:* Contiene información acerca del gestor de paquetes apt y cuales han sido los paquetes instalados.
- */var/log/fsck:* Contiene información del estado de los discos, se recoge comprobaciones de disco periódicas. No está en uso o no se ha configurado para recoger información en los logs.
- */var/log/gdm3:* Contiene los logs relativos al gestor de inicio de sesión por defecto en el entorno de escritorio Gnome. No se encuentra en uso, probablemente por una actualización a Lightdm.
- */var/log/lightdm:* Contiene los logs relativos al gestor de inicio de sesión Lightdm, se trata de un gestor de inicio de sesión genérico.
- */var/log/munin:* Contiene los logs relativos a la herramienta de monitorización Munin.

El resto de carpetas se encuentran en desuso debido a que se encuentran vacias y por tanto corresponden a paquetes que ya no están instalados o bien no se han configurado para almacenar logs.

Identificación del tipo de información que almacenan los logs alojados en la raíz /var/log

- */var/log/alternatives.log:* Contiene información sobre los paquetes alternativos propuestos por apt.
- */var/log/auth.log*: Contiene información acerca del proceso de autenticación y sesión que siguen los usuarios en el sistema, tanto de forma local como remota.
- /var/log/*daemon.log*: Contiene información acerca del sistema operativo en ejecución y los demonios de aplicación en funcionamiento.
- /var/log/*debug.log:* Contiene información de depuración de distintos programas que se ejecutan en el equipo.
- /var/log/*dpkg.log:* Contiene los cambios de paquetes realizados en el equipo.
- */var/log/kern.log*: Contiene información crítica acerca del kernel del sistema operativo.
- */var/log/messages*: Contiene información que ha ocurrido de forma genérica en el sistema. No contiene mensajes críticos ni debug

- */var/log/mail.log*: Contiene información acerca del servidor de correo del sistema.
- */var/log/syslog.log*: Contiene información sobre distintos procesos o programas, también puede aparecer mensajes sobre el kernel o sistema operativo.
- */var/log/user.log*:.Contiene información sobre los niveles de ejecución relativos a los usuarios.
- */var/log/wtmp.log*:.Contiene información sobre los accesos de los usuarios al equipo.
- */var/log/xorg.log*: Contiene información sobre el servidor gráfico.

2.3.Casos de estudio y herramientas previstas

Casos de estudio propuestos respecto al material recibido

Con la información recibida se pueden realizar distintos tipos de estudio. Pues se contiene amplia información del funcionamiento del equipo.

Se proponen los siguientes casos de estudio:

1. Estudio acerca del comportamiento de los usuarios: Debido a que se trata de un servidor que tiene salida a internet y que además está pensando para que un gran número de usuarios accedan a él, un posible estudio es analizar cómo se comportan los usuarios en el equipo cuando acceden a él.
2. Estudio acerca de la exposición del equipo a internet: Con los logs recibidos y sabiendo que los usuarios acceden a él a través de internet, un posible estudio es comprobar cuál es el impacto que recibe este servidor al estar expuesto a internet de forma permanente y cuáles son los posible ataques que puede haber recibido.
3. Estudio acerca de las aplicaciones que se encuentran en equipo: se puede abordar el estudio acerca de cuáles son los programas que se encuentran funcionamiento, de cara a realizar un informe de seguridad, averiguar si de los programas en funcionamiento, alguno tiene algún tipo de actividad sospechosa o comprobar si la integridad del equipo ha sido comprometida.

Necesidad de herramientas

Como se ha mostrado anteriormente se dispone de una gran cantidad de logs relativos a distintos funcionamientos que ocurren en el equipo. Por lo que se descarta de pleno una revisión totalmente manual de los ficheros debido tanto a sus dimensiones como a la cantidad de ellos. Por ello se precisa de algún tipo de herramientas o scripts que permitan evaluar de forma eficaz el contenido que está almacenado en los logs del equipo.

Para ello se propone lo siguientes casos:

1. Desarrollo de herramientas.
2. Uso de herramientas ya desarrolladas.

Una posibilidad es el desarrollo de herramientas a medida capaces de analizar los logs o parte de los logs disponibles, pero supondría ocupar gran parte del tiempo desarrollando una aplicación, restando tiempo al análisis.

O por otro lado utilizar herramientas que ya se encuentren disponibles con un curva de aprendizaje probablemente menor que el tiempo de desarrollo permitiendo dedicar más tiempo al análisis.

Acotación del estudio

Se considera que dentro de los casos de estudio propuestos, por predilección del alumno, resulta más interesante el estudio del comportamiento que se realiza dentro del servidor por parte de los usuarios, así como el estudio de la exposición del equipo a internet.

Respecto al estudio de las herramientas y programas que se encuentran ya instalados en el equipo, se considera que se dispone de información parcial, pues no se dispone de una imagen de memoria del equipo ni acceso con permisos de súper-usuario al equipo.

Por lo que el estudio que se realiza estará basado en los puntos 1 y 2 Estudio acerca del comportamiento de los usuarios y Estudio acerca de la exposición del equipo a internet, se piensa además que este tipo de estudios son complementarios añadiendo así riqueza de un estudio a otro, pues un acceso al equipo siempre va a ser realizado por un usuario y eso denota un comportamiento y una exposición a internet también y por supuesto una vez que un usuario ha iniciado sesión se observa el comportamiento del usuario.

Logs seleccionados para realizar el análisis

Una vez catalogado y estudiado el material que se dispone, el siguiente paso es seleccionar cuales de todos los logs de los que se tiene es importante analizar con detenimiento. Por cantidad de información y por cantidad de logs disponibles sin duda el más interesante es:

/var/log/auth.log

Se dispone de una gran cantidad de logs (356) y además contiene información que enmarca los dos casos de estudio, la autenticación de usuarios que se produce tanto a nivel local como a través de internet, además contiene información de las múltiples conexiones que se realizan al equipo, dando así una buena imagen de cuál es la exposición del equipo a internet.

Búsqueda de herramientas

Antes de desarrollar herramientas se ha procedido a evaluar cuál es el estado del mercado actual, buscando alguna herramienta que pueda servir para analizar los logs */var/log/auth.log* y en el caso de que no se disponga de alguna o resulte incompatible se procedería al desarrollo de una herramienta a medida que cubra la necesidad de analizar y procesar los logs.

2.3.1. Splunk

Splunk se trata de una plataforma de software comercial para analizar el estado de una o varias máquinas en tiempo real.

Está disponible como herramienta *On-Premise* y *SaaS* por lo que existen dos versiones, una versión para ser instalada de forma local en los equipos que se deseen analizar y también existe la posibilidad de forma SaaS en la nube para que reciba los logs de las máquinas.

El objetivo de Splunk es convertir y procesar los datos de las computadoras y máquinas para identificar patrones, estadísticas, incidencias y que pueda ser utilizado en operaciones de inteligencia de negocio. Se trata de una de las herramientas de análisis de logs más populares bajo el modelo de licencia comercial, es utilizada por gigantes comerciales como Cocacola por lo que se trata de un producto maduro y probado.

Utiliza como lenguaje de búsqueda SPL, search processing language que permite realizar búsqueda de texto completo.

Dispone de la siguiente arquitectura:

1. **Nodos**: Serán las máquinas distribuidas donde se realizaran las consultas.
2. **Splunk**: Modulo encargado de realizar el indizado y procesado de la información en tiempo real.
3. **DataStore**: Almacena y comprime la información de forma cruda, es decir inalterada.
4. **Bundles**: Guarda la información refinada.
5. **Modules**: Módulos que permiten añadir funcionalidades adicionales.

Consta de las siguientes funcionalidades clave:

1. **Indizado:** Permite almacenar la información de los logs de diferentes fuentes para que luego sea recuperable por los distintos servicios que componen Splunk.
2. **Búsqueda de información:** Permite buscar de forma flexible eventos o errores en la platafoma que monitoriza Splunk.
3. **Contrastado de la información**: Permite exportar los datos obtenidos a distintas plataformas para que puedan encontrarse en una organización para que los utilicen como fuente de datos refinada.
4. **Alertas**: Permite definir un umbral o reglas que cuando se cumplan en base a los datos de los logs generen una alerta o acción correctiva.
5. **Reportes:** Dispone de una gran cantidad de plugins que permiten construir gráficos y estadísticas adaptables a distintas formas.
6. **Machine Learning**: Consta de un módulo de inteligencia artificial que permite en base al histórico de los logs analizados inferir futuros comportamientos de las máquinas.

No obstante no todas las funciones están disponibles para todos los tipos de usuarios, al disponer de un modelo de negocio de software comercial existen distintos planes con distintos precios en base al plan elegido y sus funciones como de la cantidad de Gigabytes procesados, también dispone de versión FREE pero no dispone de todas las características:

Precios a día 13/08/2018 desde la web oficial de Splunk.

Plan	Límite usuarios	Límite de datos al día	Cloud	On Premise	Indizado	Análisis tiempo real	Escalable	Soporte Oficial	Precio
Free	1	500mb	No	Si	Si	Si	No	Comunidad	$0
Light	5	20Gb al día	Si	Si	Si	Si	No	Base	$87 Gb/mes
Splunk Cloud	Ilimitado	No	Si	No	Si	Si	Si	Standar	Requiere contacto
Enterprise	Ilimitado	No	No	Si	Si	Si	Si	Premium	$173 Gb/mes

Tabla 4-Comparativa de precios Splunk

Nota: También dispone de versiones de prueba de 30 días.

2.3.2. Sumo Logic

Sumo Logic se trata de una plataforma de código cerrado que se ofrece como solución empresarial para almacenar y analizar información sobre distintas máquinas a través de sus logs.

Su modelo de negocio está basado en SaaS por lo que no dispone de versión On-Premise.

La arquitectura esta por tanto situada en la nube de Sumo Logic y realiza las siguientes funcionalidades clave:

- Recolección de datos centralizada.
- Monitorización y visualización.
- Predicción y optimización.
- Alertas y notificaciones.
- Búsqueda e investigación.

Dispone de diversas aplicaciones preconfiguradas para integrar como: Windows, Apache, Tomcat, Mongodb, Google Apps entre otras.

También se destaca por aplicar en su infraestructura cloud modelos de seguridad certificados que pueden encontrarse aquí [18]

Finalmente las siguientes tablas recogen sus tarifas a día 13/08/2018:

Plan	Soporte	DPM* FREE	Capabilities	Precio Base Mes 3Gb	Precio Base Mes 5Gb	Precio Base Mes 10Gb	DPM* adicionales
Sumo Logic Free	Community	-	Free capabilities	-	-	-	-
Professional	Sumo Logic Support	1000	Pro capabilities	$297	$495	$990	$15 cada 1000 DPM
Enterprise	Sumo Logic Support	1000	Enterprise capabilities	$495	$825	$1650	$15 por 1000 DPM

Tabla 5- Comparativa de precios Sumo Logic

DPM: Data points per minute.

Capabilities:

Free	Professional	Enterprise
Data Collection (any Log source)	All Sumo Logic Free Capabilities	All Professional Capabilities
Powerful Ad Hoc Search	Real Time Alerting	Search API Access
LogReduce & LogCompare Analytics	Data Forwarding & Webhook Integrations	Single Sign-on Integration
Predictive Analytics & Outlier Detection	Advanced Search Performance	Security Analytics App Framework
Live Tail for Streaming Logs	Collector Management API	PCI Compliance App Framework
Historical & Live Streaming Dashboards	Sumo Logic Apps	Free Metrics, Up To 5,000 DPM
PCI, SOC, CSA, ISO, HIPAA Certifications	Free Metrics, Up To 5,000 DPM	Variable, Multi-Year Retention
	Variable, Multi-Year Retention	

Tabla 6- capabilities Sumo Logic

Nota: También dispone de versiones de prueba de 30 días y planes personalizados.

2.3.3. Loggly

Se trata de una solución software para el análisis de logs de la empresa Solarwings. Se trata de un modelo de software comercial cerrado cuyas características principales son:

- **Visor pro-activo**: Monitor que está caracterizado por observar el comportamiento de las máquinas monitorizadas, registrando comportamientos sospechosos. Se trata de una prevención de amenazas.
- **Detección de problemas**: Análisis del comportamiento de las aplicaciones y sus componentes para detectar fallos en la arquitectura.
- **Integración con DevOps**: Integración con aplicaciones y herramientas de terceros para el uso de la información almacenada en Loggly. Integración con Slack, HipChat, GitHub, Jira, New Relic entre otros.
- **Análisis de datos y reportes**: Dispone de diferentes gráficos y sistemas de reporting para facilitar la investigación y el análisis.

Sus tarifas a día 13/08/2018 son las siguientes:

Plan	Soporte	Volumen	usuarios	Precio al mes
Free	Community	200mb al día	1	0
Standar	Email support	1 Gb al día	3	$79
Pro	Priority support	20Gb al día	5	$199
Enterprise	Technical success management	Variable	Ilimitados	$349

Tabla 7- Precios Loggly

Características por plan:

Free	Standar	Pro	Enterprise
Centralized log management	Up to 3 users	Up to 5 users	Unlimited users
Automated log summaries	Up to 3 source groups	Up to 5 source groups	Unlimited source groups
Search & filters	Built-in email alerting	Alerts: Email, webhook (use with PagerDuty, HipChat, Slack, VictorOps)	Unlimited derived field rules
Single user	Fast monitoring with charts & dashboards	Custom derived fields: up to 10 rules	Variable custom volume (up to TBs)
	Direct access to support team	API access	Custom retention periods
		Archive to Amazon S3	Technical success management
		Peak Overage Protection	Loggly Live Tail
			GitHub Integration
			JIRA Software integration
			Anomaly Detection
			Federated identity management

Tabla 8- Características por plan Loggly

2.3.4. PaperTrails

PaperTrails se trata de una herramienta para el manejo de logs de forma centralizada. Permite realizar búsquedas y análisis de logs de distintas máquinas.

Dispone de una interfaz sencilla que permite realizar las siguientes características:

- Logs unificados: Permite recolectar logs de diferentes máquinas a la vez.
- Modos de búsqueda: Permite realizar búsquedas a través del buscador integrado que dispone o bien a través de la línea de comando o la API.
- Detección de tendencias: Capacidad de generar alertas en base a tendencias
- Búsqueda rápida: Sistema de búsqueda que permite consultar varios logs a la vez.
- Integración con otros servicios a través de su API.

Sus tarifas están elaboradas en base al número de datos consumidos al mes siendo las siguientes:

Precio al mes	Volumen de datos al mes
Free	100MB
$5	1GB
$12	2GB
$22	4GB
$42	8GB
$82	16GB
$138	25GB
$185	50GB
$325	100GB
$485	250GB
$875	500GB
$1460	1000GB
$1945	1500GB

Tabla 9- Precio GB PaperTrails

El precio también varía en función del tiempo que los logs permanecen en la plataforma por lo que el precio final también se ve afectado por esta variable. Sin embargo no se añade una tabla adicional por disponer de un método poco claro acerca de su tarifación, en caso de dudas consultar el siguiente enlace [19]

2.3.5. GrayLog

Se trata de software OpenSource basado en ElasticSearch, MongoDB, Java y Scala para el análisis y recolección centralizada de logs. Dispone de dos modelos: El modelo OpenSource en el que el usuario se encarga del producto y el modelo empresarial que ofrece soporte técnico.

Consta de las siguientes funcionalidades clave:

1. **Recolección y procesado de logs**: Graylog ofrece la capacidad de leer distintos tipos de logs de diferentes fuentes para ser tratados de forma de centralizada, asi como integración con directorios LDAP.
2. **Dispone de API REST**, para interactuar con la aplicación.
3. **Capacidad de transformar los datos** para que se puedan utilizar por otras aplicaciones como fuente de datos refinada.
4. **Análisis e investigación**: Capacidad para buscar y analizar información en diferentes de fuentes de datos a la vez con una sola consulta.
5. **Gráficos y estadísticas**: Dispone de múltiples gráficos para analizar los datos de forma sencilla accesibles mediante una interfaz web.
6. **Alertas y contramedidas**: Capacidad de crear disparadores y acciones que se despliegen cuando se produce un evento en los logs.

Su arquitectura es la siguiente:

1. **Server nodes**: Son los encargados de leer la información de las máquinas monitorizadas y enviar los logs a las máquinas de procesado que serán los nodos Elastic Search.
2. **Elasticsearch nodes**: Almacena y procesa todos los logs y mensajes recibidos por parte de los Server Nodes.
3. **MongoDB**: Actúa como almacén metadatos.
4. **Web Interface**: Interfaz con la que el usuario visualiza los datos recolectados y procesados.

La siguiente tabla comparativa compara el modelo Open Source frente al modelo Enterprise que dispone GrayLog. Respecto al precio del modelo Enterprise es necesario ponerse en contacto con la empresa.

Características	Modelo Open Source	Modelo Enterprise
Extended Log Collection Using Sidecar	SI	SI
Scalable Log Collection	SI	SI
Log Data Enrichment	SI	SI
Simple UI for Administration	SI	SI
Graphical Log Analysis	SI	SI
Reporting Content Packs	SI	SI
Alerts & Triggers	SI	SI
REST API	SI	SI
Free Marketplace of Extensions	SI	SI
LDAP Integration	SI	SI
Views	NO	SI
Offline Log Archival	NO	SI
User Audit Logs	NO	SI
Implementation Jumpstart	NO	SI
Technical Support	10 Tickets al año	Soporte ilimitado

Tabla 10- Características GrayLog

2.3.6. Elastic Stack

Elastic Stack se trata de un proyecto OpenSource multiplataforma que contempla una serie de productos y soluciones que tienen el objetivo de facilitar al usuario tratar la información. Sus funciones básicas son extraer, buscar, analizar, visualizar datos en tiempo real o de forma asíncrona.

Este proyecto está basado en la separación de responsabilidades sobre distintas capas del producto. Por lo que se trata de un proyecto que está pensando para ser distribuido en distintas funciones y distintas computadoras.

Elastic Stack se puede implementar de distintas maneras tanto de forma local en una sola computadora como de forma distribuida en distintas computadoras o bien mediante un modelo SaaS (Software as a service).

Generalmente es conocido como ELK, que son las siglas de las principales capas que componen la estructura del proyecto siendo las siguientes:

1. ElasticSearch
2. LogStash
3. Kibana

ElasticSearch, ES

Se trata del componente principal de Elastic Stack cuya principal responsabilidad es catalogar y almacenar información para poder ser recuperada después de forma flexible.

Por lo que se trata de un motor de búsqueda OpenSource, escalable y distribuido que está basado en Apache Lucene, aunque emplea su propio lenguaje de consulta denominado Query DSL para recuperar la información almacenada. Al estar basado en Apache Lucene busca minimizar los tiempos de consulta con el mismo hardware respecto a utilizar SQL convencional.

Ofrece una API, que es de tipo RESTful Json y clientes para distintos lenguajes de programación de forma oficial: Java, Python, .NET, SQL y PHP. Por otro lado al ser OpenSource también dispone de una comunidad de usuarios que también ofrece clientes y soporte para otros lenguajes de programación.

Está compuesto por repositorios, que es donde ElasticSearch almacena la información en forma de documentos JSON donde cada documento es una correlación de clave-valor.

Su fase principal por tanto es la de almacenar y catalogar con un índice la información que está sin tratar, en el formato descrito anteriormente, ficheros JSON, para que sea fácilmente recuperable por los métodos descritos anteriormente.

Ofrece además una interfaz para trabajar con los datos de distintas plataformas de Big Data como Hadoop y Spark mediante conectores como ElasticSearch-Hadoop (ES-Hadoop).

La primera versión de ElasticSearch aparece en el año 2010 y debido a sus características se ha convertido en el motor de búsqueda de texto completo más popular y se ha convertido en casi un estándar para analizar logs del sistema. Por lo que sus características principales son:

1. **Rápido**: Debido al índice basado en repositorios, es capaz de realizar búsquedas de texto completo incluso en colecciones de documentos amplias.
2. **API flexible**: Ofrece una interfaz RESTful simple basada en HTTP.
3. **Soporte amplio para distintos desarrollos:** Debido a la amplia de lenguajes soportados de forma oficial.
4. **Plugins:** ElasticSearch se puede complementar de forma sencilla con distintos plugins para ampliar su funcionalidad.
5. **Documentación:** Debido a su popularidad se dispone de una amplia documentación y ejemplos tanto de forma oficial como de la comunidad de desarrolladores que dispone.
6. **Escalable y distribuido:** Está pensado para ser utilizado con distintos entornos.
7. **Índices cercanos a tiempo real:** El tiempo de indexado por documento es muy bajo por lo que permite realizar lecturas cercanas al tiempo real sobre eventos que están ocurriendo en el momento.
8. **Conectores con otros estándares de Big Data:** Dispone de conectores para ampliar las funcionalidades de Elastic Search como Hadoop.
9. **OpenSource:** Acceso al código y capacidad de adaptar el código a necesidades concretas.

Logstash

Este componente de software perteneciente a Elastic Stack se encarga de procesar la información en crudo para que pueda ser consumida y almacenada por un motor de búsqueda en concreto, generalmente será ElasticSearch. Está estructurado en forma de tubería o pipeline por lo que LogStash dispone de 3 partes diferenciadas:

Entrada (inputs): Esta parte dispone de los conectores y la interfaz necesaria para que pueda recibir la información en crudo de distintas fuentes. Existe una multitud de plugins para distintos servicios y datos.

Procesado y filtros (parse): Su objetivo es procesar la estructura de la información para que sea convertida a un formato comprensible y manejable por un motor de búsqueda. El modo en que procesa esta información está basada en filtros en la que cada filtro tiene una función distinta para convertir la información.

Salida (output): Se trata de la configuración de salida de la información que han procesado los filtros. Por defecto la salida es ElasticSearch pero dispone de la capacidad de ofrecer el formato para otras plataformas o servicios.

Kibana

Se trata de la interfaz gráfica que hará la función de monitor. Por tanto se encargará de leer la información contenida en Elastic Search para después presentarla a través de una interfaz web donde se visualiza de forma fácil el contenido almacenado en el motor de búsqueda.

Dispone de multitud de gráficas, métricas, histogramas y visualizaciones configurables mediante plugins que permiten analizar desde distintos puntos de vista la información.

Está construido a través de node.js y por tanto se trata de un cliente web multiplataforma opensource con la capacidad de ser ejecutado en distintas plataformas.

2.4. Conclusiones acerca del estado del arte

Tras analizar con detalle cuales son las herramientas que se están utilizando en el mercado se ha decidido utilizar alguna de las analizadas, debido a que se llega a la conclusión de que se trata de un entorno estudiado y maduro en el que reinventar la rueda quizá no es la mejor forma de abordar el problema. De esta manera con el ahorro de tiempo y esfuerzo que se va a dedicar al desarrollo, se invierte así más tiempo en aprender una herramienta que se utiliza en el mercado y permita enriquecer el trabajo pudiendo ser replicado y trasladado a otra empresa, entorno o proyecto. Por lo que el desarrollo desde 0 de una herramienta a medida queda descartado y se procederá a elegir una herramienta a utilizar en los siguientes puntos con el fin de analizar los logs recibidos.

Capítulo 3: Requisitos

3.1.Planteamiento para resolver el problema

Para la realización del proyecto se ha recogido una serie de requisitos que debe de cumplir el software utilizado durante el proyecto.

Se precisa de lo siguiente:

1. Software de tratamiento de logs, de ahora en adelante *manejador de logs*.
2. Software y herramientas donde va a ejecutarse el manejador de logs, de ahora en adelante *sistema operativo.*
3. Hardware donde va a alojarse el *sistema operativo*.

3.2.Definición de requisitos

Para definir los requisitos se utiliza una serie de tablas dónde cada requisito ocupará una tabla, por lo tanto existirá una tabla por cada requisito. La estructura de cada tabla se describe en el siguiente punto.

Estructura de tabla para los requisitos

Para definir los requisitos del proyecto se va a utilizar el siguiente modelo de tabla:

ID
NOMBRE
DESCRIPCION
CATEGORIA

Tabla 11- Modelo de tabla para los requisitos

Que contiene los siguientes campos:

1. ID: Será una cadena de texto empezando por RY-XX donde Y será una letra indicando el tipo de requisito que se trata y XX serán valores numéricos indicando el número.
   - Los requisitos que definen el *manejador de logs* usarán la letra S.
   - Los requisitos que definen el *sistema operativo* usarán la letra O.
   - Los requisitos que definen el *hardware* recibirán la letra H.
2. Nombre: Contendrá el nombre del requisito.
3. Descripción: Contiene una breve descripción del requisito.
4. Categoría: Contiene la categoría a la que pertenece el requisito siendo una de las posibles {*Manejador de logs, Sistema Operativo, Hardware*}

3.2.1. Requisitos para el manejador de logs

RS-01
Lectura automática de logs
El software elegido debe de ser capaz de leer los logs de forma automática.
Manejador de logs

Tabla 12- Requisito RS-01

RS-02
Tratamiento de logs
El software elegido debe de ser capaz tratar los logs leídos.
Manejador de logs

Tabla 13- Requisito RS-02

RS-03
Open Source
El software elegido debe de ser Open Source para facilitar la flexibilidad, compresión y estudio del software.
Manejador de logs

Tabla 14- Requisito RS-03

RS-04
Coste
El software elegido debe de tener el mínimo coste debido a que se trata de tareas académicas.
Manejador de logs

Tabla 15- Requisito RS-04

RS-05
Despliegue
El software elegido debe de tener la capacidad de ser instalado en una computadora de forma local.
Manejador de logs

Tabla 16- Requisito RS-05

RS-06
Gráficos y estadísticas
El software elegido debe de tener la capacidad de mostrar gráficas y estadísticas sobre los logs.
Manejador de logs

Tabla 17- Requisito RS-06

RS-07
Escalable
El software elegido debe de ser escalable en cuanto a software como hardware.
Manejador de logs

Tabla 18- Requisito RS-07

RS-08
Documentación y soporte
El software elegido debe de estar correctamente documentado y disponer de forma accesible información abundante.
Manejador de logs

Tabla 19- Requisito RS-08

RS-09
API
El software elegido debe de tener una API con la que poder interactuar de forma sencilla.
Manejador de logs

Tabla 20- Requisito RS-09

RS-10
Replicable
El software elegido debe de permitir que otras personas repliquen el estudio que se realice con él de forma sencilla.
Manejador de logs

Tabla 21- Requisito RS-10

RS-11
Plugins
El software elegido debe de poder ampliar su funcionalidad mediante plugins.
Manejador de logs

Tabla 22- Requisito RS-11

3.2.2. Requisitos para el hardware

RH-01
Arquitectura x86
El software manejador de logs elegido debe de dar soporte la arquitectura de procesadores x86
Hardware

Tabla 23- Requisito RH-01

RH-02
RAM
Mínimo 8GB
Se trata de una recomendación para pruebas de concepto en máquinas locales, para un sistema en producción se necesita más memoria RAM.

Tabla 24- Requisito RH-02

RH-03
CPU
Mínimo 2 cores
Se trata de una recomendación para pruebas de concepto en máquinas locales, para un sistema en producción se podría necesitar más cores.

Tabla 25- Requisito RH-03

3.2.3. Requisitos para el sistema operativo

RO-01
Soporte para el manejador de logs
El sistema operativo debe de ser capaz de ejecutar el software elegido para el manejador de logs.
Sistema operativo

Tabla 26- Requisito RO-01

3.2.4. Elección de herramientas

Este punto trata acerca de la justificación que se ha realizado para elegir las herramientas que se han utilizado en el proyecto. En las tablas siguientes se representa en rojo que no se cumple un requisito, mientras que en verde indica que lo cumple. Amarillo indica problemas para cumplirlo.

3.2.5. Trazabilidad de los requisitos para el manejador de logs

En este punto se recoge la matriz de trazabilidad entre los requisitos definidos en el punto 4 y las herramientas que se recogen en el punto 3.

Software\Requisitos	RS-01	RS-02	RS-03	RS-04	RS-05	RS-06	RS-07	RS-08	RS-09	RS-10	RS-11
Splunk											
Sumo Logic											
Loggly											
PaperTrails											
GrayLog											
Elastic Stack											

Tabla 27- Matriz de trazabilidad para entre requisitos y el manejador de logs

Justificación de la solución elegida para el manejador de logs

Se ha elegido como solución para abordar el problema Elastic Stack porque se adecua perfectamente a los requisitos elegidos. Se ha elegido además Elastic Stack frente a GrayLog debido a que GrayLog se trata de un desarrollo que se nutre de Elastic Stack por lo que se ha decido ir a la fuente original del proyecto.

Respecto a la elección de Elastic Stack como solución Open Source frente a otros competidores de software comercial como Splunk que se trata del modelo que mejor sale parado de tipo software cerrado, no se ha elegido Splunk debido a que la versión gratuita está muy limitada en cuento a lo que se desea procesar que en este caso supera el límite de los 500MB, además de que el soporte y el acceso a la documentación no es accesible para un usuario de tipo FREE por lo que hace que si se desea realizar el proyecto con un tipo de software comercial obteniendo beneficios adicionales para poder realizarlo dificultan la replicación del proyecto por otras personas interesadas en replicarlo. Se piensa que este tipo de soluciones son buenas para problemas o situaciones de carácter empresarial donde se precisa el soporte de una empresa, para un proyecto de carácter académico se considera que se encuentra sobredimensionado elegir una solución de carácter comercial adquiriendo licencias de uso, además de que se pierde el carácter didáctico de conocer como está construida la infraestructura y de la capacidad de modificarse si se requiere.

Por lo que el software elegido para tratar los logs será Elastic Stack en su versión actual a día 14/08/2018. Siendo la versión 6.3.

3.2.6. Matriz de trazabilidad entre distintos sistemas operativos y Elastic Stack

Utilizando la documentación disponible sobre Elastic Stack se ha elaborado la siguiente matriz de trazabilidad que indica el soporte a los distintos sistemas operativos con cada versión Elastic Stack 6.3.X siendo la última versión a día 14/08/2018.

	Elasticsearch 6.3.x	**Logstash 6.3.x**	**Filebeat 6.3.x**
CentOS/RHEL 6.x/7.x			
Oracle Linux 6/7 RHEL			
Ubuntu 14.04			
Ubuntu 16.04			
SLES 11 SP4			
SLES 12			
openSUSE Leap 42			
Windows Server 2012			
Windows Server 2016			
Debian 7			
Debian 8			
Debian 9			
Solaris			
Amazon Linux			

Tabla 28- Matriz de trazabilidad entre distintos sistemas operativos y ElasticStack

3.2.7. Trazabilidad de los requisitos por sistemas operativo

	RO-01
CentOS/RHEL 6.x/7.x	
Oracle Linux 6/7 RHEL	
Ubuntu 14.04	
Ubuntu 16.04	
SLES 11 SP4	
SLES 12	
openSUSE Leap 42	
Windows Server 2012	
Windows Server 2016	
Debian 7	
Debian 8	
Debian 9	
Solaris	
Amazon Linux	

Tabla 29- Matriz de trazabilidad de requisitos por sistema operativo

Solución elegida para el sistema operativo, justificación

Existen varios sistemas operativos que cumplen la matriz de requisitos del sistema operativo siendo los siguientes:

CentOS/RHEL 6.x/7.x, Ubuntu 14.04, Ubuntu 16.04, SLES 12, Windows Server 2012, Windows Server 2016, Debian 8, Debian 9

Al no existir ninguna ventaja respecto uno de otro se ha elegido Debian 9 por disponer de una máquina para probar con Debian 9 ya instalado.

3.2.8. Justificación del uso de Java

Para poder ejecutar parte de la arquitectura de Elastic Search se precisa Java, en los siguientes puntos se describe la elección de la versión de Java.

Trazabilidad entre Java y ElasticSearch/LogStash

Este punto recoge distintas máquinas virtuales de Java junto con los elementos que precisan Java como dependencia.

	ElasticSearch 6.3	Logstash 6.3
Oracle JVM 1.7u55+*		
Oracle JVM 1.8u60+		
Oracle JVM 9		
Oracle JVM 10		
IcedTea OpenJDK 1.7.0.55+*		
IcedTea OpenJDK 1.8.0.111+		
OpenJDK 9		
OpenJDK 10		
Azul Zing 16.01.9.0+		
IBM JVM 7, 8, 9 & 10		

Tabla 30- Matriz de trazabilidad entre Java y ElasticSearch/LogStash

Como muestra la gráfica anterior las versiones de la máquina virtual de Java soportadas son la versión 8 y 10.

Solución elegida para Java, justificación

Se ha elegido la versión Open Source de la máquina virtual de Java OpenJDK 1.8 por comodidad ya que se encontraba instalada en el sistema y es compatible con las versiones propuestas.

3.2.9. Elección de navegador web

Para que Kibana pueda mostrar los datos que se recogen en ElasticSearch se utiliza un navegador para consultar sus datos. La matriz de compatibilidad entra navegadores es la siguiente:

	Kibana 6.3
IE9	
IE11+	
Firefox	
Chrome	
Safari	

Tabla 31- Matriz de compatibilidad entre navegadores web y Kibana

Justificación de la elección de navegador web

Se ha elegido Firefox como navegador para realizar las pruebas, debido a que se encuentra instalado en la máquina que se va a utilizar y es multiplataforma. Pero cualquier otro navegador del listado sería válido, excepto las versiones antiguas de IE, que no están soportadas.

3.2.10. Solución elegida para el hardware, justificación

La máquina que tiene Debian 9 instalado dispone del hardware necesario para realizar pruebas de concepto pero no para ser una máquina de producción por lo que se ajusta a los requisitos mínimos requeridos. Siendo la configuración la siguiente:

CPU: Intel i5 7ª Generación con 4 cores, 64 bits

Memoria RAM: 12 GB

3.2.11. Especificaciones, resumen

Solución final, especificaciones:

CPU: Intel i5 7ª Generación con 4 cores, 64 bits.

Memoria RAM: 12 GB.

Sistema operativo: Debian 9 Strech.

Máquina virtual de Java: OpenJDK 1.8.

Navegador: Firefox 61.1.02

Manejador de logs: Elastic Stack.

- ElasticSearch 6.3
- LogStash 6.3
- Kibana 6.3
- FileBeat 6.3

Capítulo 4: Diseño y configuración de herramientas

4.1.Diseño del sistema y alternativas

Este punto trata acerca del diseño que se ha realizado con Elastic Stack para abordar el problema que plantea el proyecto. Para ello se ha elaborado un plan estratégico breve sobre las distintas fases que componen el diseño.

4.1.1. Estrategia a seguir

Los siguientes pasos recogen de forma breve cuál es el proceso que se pretender seguir para analizar los logs sobre el sistema Guernika.

1. **Generación de logs**: Esta parte concierne al momento en que los logs son generados. En el caso evaluado los logs son generados de forma automática por el sistema Guernika.
2. **Almacenamiento de logs**: Trata acerca del momento en el que los logs generados son almacenados. En el caso evaluado son almacenados de forma automática por el sistema Guernika en el directorio */var/log/*
3. **Selección de logs a analizar**: Se trata del proceso de selección de logs a analizar que se encuentran en el directorio */var/log/* tarea que realizará *FileBeat*.
4. **Tratamiento y depuración de logs** Trata sobre cómo los logs son consumidos y leídos por el manejador de logs para que la información que se encuentra en ellos sea depurada y clasificada para su posterior uso. Se trata del refinamiento de la información contenida en los logs a partes más pequeñas que faciliten el almacenamiento y permitan realizar búsquedas de información más fáciles. Esta parte estará realizada por parte del entramado de *ELK*. Siendo los componentes encargados de esta tarea *LogStash* y *FileBeat*.
5. **Almacenamiento refinado de los logs** Consta del proceso de almacenar de forma estructurada la información refinada obtenida de los Logs para que pueda ser consultada de forma más fácil más adelante. Este proceso será realizado por el componente de ELK, ElasticSearch.
6. **Consulta refinada de los logs** Consta del proceso de recuperar de forma lógica la información refinada almacenada sobre los Logs. Este proceso será realizado por el componente de ELK, ElasticSearch.
7. **Visualización refinada de los logs** Se trata del estudio y conclusiones de la información refinada almacenada. Para ello se utilizará el visor de ELK Kibana.

4.2.Diseño de la arquitectura

Debido a que se va a utilizar una colección de logs de forma estática y no en tiempo real, se ha propuesto realizar una arquitectura de forma local en una misma máquina, delegando las distintas partes de Elastic Stack a procesos distintos, permitiendo así el Sandboxing entre aplicaciones.

4.2.1. Esquema sobre el diseño propuesto

El siguiente esquema recoge de forma conceptual cuál es la infraestructura utilizada con ELK en base a la estrategia definida.

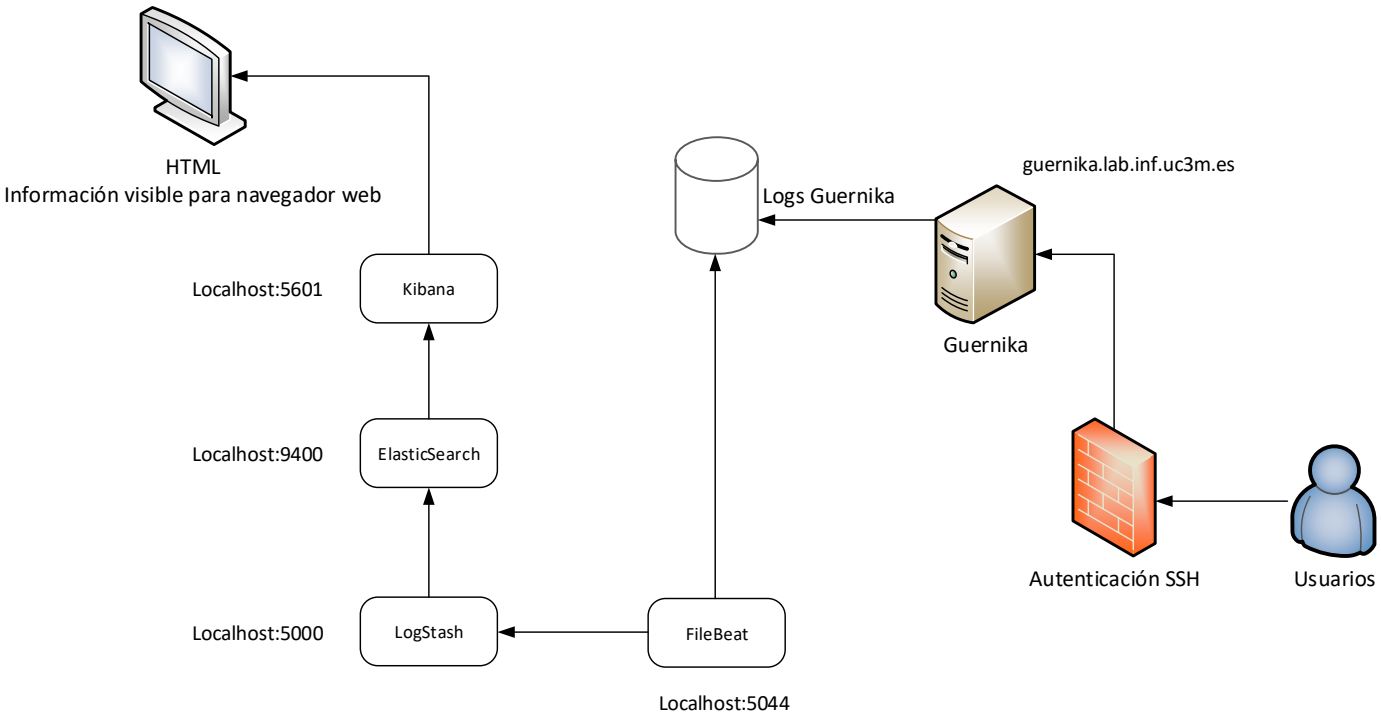

Ilustración 1- Esquema sobre el diseño propuesto

La figura mostrada muestra cuál sería el flujo de recuperación de la información por parte de ELK.

Un usuario trata de conectarse mediante SSH a Guernika bajo la url *guernika.lab.inf.uc3m.es y puerto 22* o mediante acceso físico. La diferencia entre ambos métodos es que mediante SSH la conexión se realiza a través de internet y mediante acceso físico es que se accede localmente a la máquina para realizar login.

Los intentos de autenticación tanto exitosos como no, son registrados en el sistema así como la actividad de cada usuario, mediante logs. También es registrado el funcionamiento del sistema independientemente de la actividad del usuario.

Estos logs, en el caso que ocupa son leídos por FileBeat. Y serán redirigidos hacía LogStash mediante la url localhost y puerto 5000.

LogStash se encarga de procesar los logs y transformar la información que se encuentra en los Logs.

El resultado de refinar la información en partes más pequeñas y con significado propio, es enviado a ElasticSearch mediante la url localhost y puerto 9400.

Finalmente para visualizar la información, Kibana lee la información de ElasticSearch en el puerto 9400. Esta información será visualizada mediante una url con un navegador web, mediante la url localhost y puerto 5601.

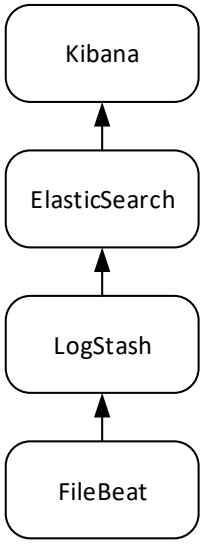

Ilustración 2- Esquemá básico sobre el sentido y orden en el que viaja la información en la arquitectura software diseñada

El diagrama anterior recoge cuál sería la sucesión y el orden de los elementos de Elastic Stack para recuperar la información. El sentido es desde abajo hacia arriba, siendo la parte más baja, la lectura de logs y la parte alta la visualización por el usuario final.

4.2.2. Alternativas sobre el diseño propuesto

La alternativa al diseño propuesto consiste en utilizar la arquitectura de forma distribuida. Con las siguientes configuraciones posibles:

1. **Un servidor cloud ejecutando ELK**: Consiste en utilizar la arquitectura propuesta completa de ELK en un servidor remoto que podría estar alojado en un entorno cloud como podría ser Amazon Web Services (AWS) o Windows Azure o bien servidores propios conectados entre sí. De esta forma se tendría un entorno escalable que permitiría tener la configuración de la arquitectura ELK bajo la demanda que se pudiera producir en el procesamiento.
2. **Distintos servidores cloud ejecutando distintos elementos de ELK**, consiste en distribuir en distintos servicios como los mencionados, partes de la arquitectura o la arquitectura completa. De esta forma por ejemplo se podría tener distintas máquinas con FileBeat de forma local, que enviaran la información a un LogStash remoto y que este a su vez enviara los datos refinados a otra máquina ejecutando Elastic Search y Kibana. Este podría ofrecer la visualización de los datos a través de Kibana en una url fija a otros equipos remotos.

4.2.3. Relación de servicios en la arquitectura propuesta

Este apartado contiene de forma detallada el diseño de los servicios de la infraestructura ELK. Contiene la relación de servicios, dirección IP, puerto, información necesaria y salida de información.

La siguiente tabla contiene el resumen de servicios que componen la infraestructura ELK:

Servicio	IP	Puerto	Procesa	Genera
Kibana	localhost	5601	Logs refinados	HTML
ElasticSearch	localhost	9400	Logs en formato JSON	Logs refinados
LogStash	localhost	5000	Logs sistema	Logs con formato JSON
FileBeat	localhost	5044	-	Redirige a LogStash

Tabla 32- Relación de servicios en la arquitectura propuesta

Nota: En este caso se ha propuesto como método de despliegue la misma máquina por lo que se trata de una relación de servicios desplegados en la misma máquina. Pudiendo separar de forma completa cada servicio en máquinas distintas.

Bastaría con sustituir la dirección IP localhost por la dirección IP de la máquina que ejecute cada servicios.

4.2.4. Diseño elegido

Se ha elegido el diseño inicial debido a que se trata de un análisis de datos estático. Es decir el procesamiento de la información sólo se realiza una sola vez por lo que arquitecturas que sean escalables no tienen una ventaja frente a un entorno más modesto pero que sea capaz de procesar la información que recibe.

Si se precisara de un entorno que necesitase estar monitorizado en tiempo real y con un gran volumen de información a procesar entonces sí que los entornos distribuidos tienen una ventaja notable, son escalables y los recursos son elásticos, es decir se consumen los que se necesiten. Por este motivo se descarta la implementación de los entornos distribuidos de los que se hablará más adelante en el capítulo líneas futuras.

4.3.Implementación del diseño

El siguiente punto trata acerca del proceso que se ha seguido para implementar el diseño elegido.

4.3.1. Instalación

El software de Elastic Stack se encuentra disponible en la web oficial:

Se ha elegido como método de instalación utilizar paquetes de tipo .deb que se pueden encontrar aquí:

- Software para ElasticSearch [20]
- Software para LogStash [21]
- Software para FileBeat [22]
- Software para Kibana [23]

Una vez descargados se procede a su instalación con:

dpkg –i elasticsearch.deb

dpkg –i logstash.deb

dpkg –i kibana.deb

dpkg –i filebeat.deb

Con esto se consigue que el software esté instalado en el sistema, el siguiente punto trata acerca de la configuración de la instalación realizada

4.3.2. Adecuar el formato de la colección de logs

Una de las primeras tareas realizadas es descomprimir los logs que se encuentran comprimidos en formato gzip. Para ello se han movido los ficheros a una carpeta y han sido descomprimidos para que FileBeat pueda acceder a ellos y sea capaz de enviarlos a LogStash para su procesado.

Ejemplo de listado de logs a descomprimir

Ilustración 3- Listado de logs a procesar

cd carpeta_logs

Donde *carpeta_logs* se trata de la carpeta donde están situados los logs para analizar.

Ejemplo de descompresión de los logs, con el comando *gunzip* en la carpeta donde se encuentren. El parámetro auth.* identifica todos los ficheros que comiencen por la cadena auth.

Ilustración 4- Descomprensión de los los logs auth.log comprimidos

Tras haber descomprimido los logs ahora se encontrarán en el formato original que tenía el servidor almacenado. La compresión se realiza para que se ahorre espacio en disco en el servidor pero de cara a analizar los logs esto impide que sean legibles de forma fácil.

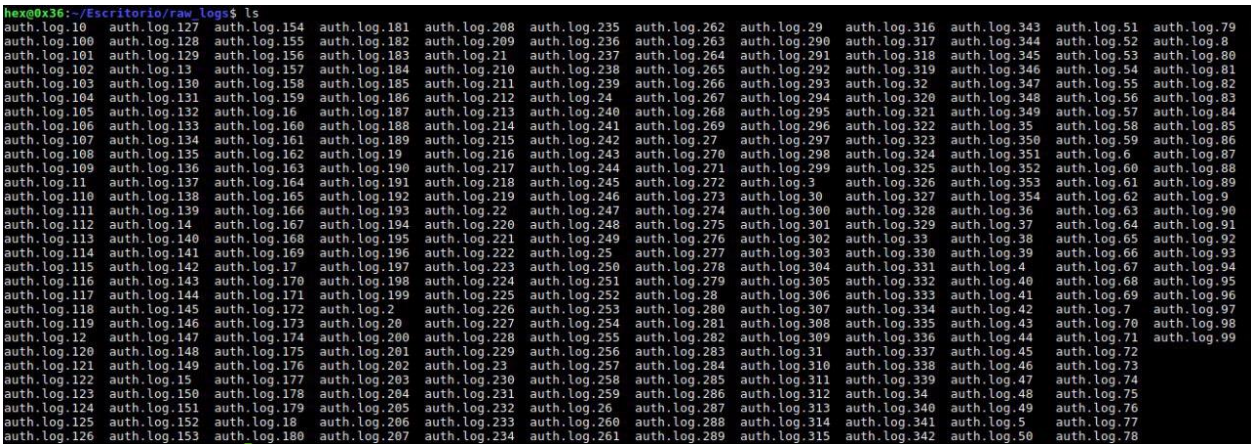

Ilustración 5- Logs descomprimidos en el formato original

4.3.3. Configuración de la instalación

Una vez se dispone de la arquitectura instalada, es necesario configurar los distintos elementos de ELK para que funcionen correctamente, para ello es necesario configurar tanto FileBeat como LogStash con ficheros de configuración personalizados.

4.3.4. Configuración de FileBeat

Para configurar Filebeat se necesita crear un fichero de configuración YAML que permita indicar que tipo de fichero debe de leer, la ruta donde debe leer los ficheros y donde depositará la salida de los logs que encuentre:

Para ello se crea un fichero con un editor cualquiera en este caso se ha utilizado *vim*:

vim filebeat_config.yml

```
filebeat.prospectors:
- type: log
  paths:
    - "/home/hex/Escritorio/raw_logs/auth.*"
output.logstash:
  hosts: ["localhost:5044"]
```

Ilustración 6- Configuración de FileBeat

Consta de los siguientes campos:

- **type**: Indica el formato del fichero que va a recibir como parámetro de entrada.
- **Paths**: Ruta en la que se encuentran los ficheros. La ruta debe de ser absoluta. En este caso se ha utilizado la expresión *auth.** para referirse a todos los logs que comiencen con su nombre la cadena de texto auth.
- **Output.logstash**: Indica la salida de los ficheros que encuentre FileBeat, en este caso será LogStash y se indica la dirección IP dónde se ejecuta LogStash y el puerto. En la arquitectura utilizada su dirreción IP es la local, motivo por el que se utiliza localhost y el puerto 5044.

4.3.5. Configuración de LogStash

LogStash permite dividir en partes más pequeñas la información que recibe a través de los logs, nos obstante por defecto no se realiza ningún tipo trabajo y es necesario configurar LogStash para que realice estas tareas.

Para entender qué es lo que se debe de evaluar con LogStash se debe de prestar atención a la estructura de los ficheros *auth.log*.

El objetivo por tanto de la configuración de LogStash es seleccionar la información que va a almacenarse. Por ello se definen los campos que debe de tratar, se trata de una anticipación al contenido que se anticipa que va a estar reflejado en el log, también la información que va a guardarse está relacionado con las búsquedas que se van a realizar en el futuro con Kibana en ElasticSearch que es quien almacenará la información que filtre LogStash.

Selección de la información:

- **Fecha y hora**: Se desea guardar cuál es la marca de tiempo de las entradas del fichero, esto va a permitir en el futuro filtrar las búsquedas por franjas de tiempo y tener una visión temporal de los sucesos ocurridos.
- **Usuario**: Se guardará el nombre de usuario que está relacionado con la entrada del log, esto permitirá identificar la entrada o múltiples entradas de distintos logs a través del nombre de usuario que se almacene.
- **IP**: Guardar la dirección IP desde la que se accede a la máquina permitirá trazar el origen de las conexiones de los usuarios.
- **Estado de la autenticación**: Conocer el estado de la autenticación que se ha producido en la conexión permitirá identificar posibles comportamientos maliciosos.
- **Método de autenticación**: Permitirá conocer que forma utiliza un usuario para identificarse frente a Guernika.
- **Superusuario**: Es interesante recoger y trazar el uso del comando sudo que permite realizar tareas con permisos de administrador, así como conocer el uso del usuario administrador denominado *root*.
- **Comando ejecutado**: Conocer el comando ejecutado por un usuario permitirá conocer las intenciones y comportamientos que se deseen trazar en el futuro.

- **Ruta**: Conocer desde dónde se ejecuta un comando permitirá conocer cuál puede ser la intención de un usuario.
- **Geolocalización**: A través de la geolocalización de los usuarios se permitirá conocer en qué punto del mundo se realizó la conexión segmentando así las conexiones por regiones.

Para poder realizar esta segmentación y filtrado de la información LogStash permite utilizar un módulo personalizable denominado Grok. La configuración será aplicada por tanto a Grok, con el fin de recoger la información de la lista anterior. Su configuración es la siguiente:

En el apartado input se configura cual será el origen de la información que recibe LogStash. Se define el puerto sobre el que recibirá los datos de entrada y en qué dirección IP debe hacerlo.

```
input {
  beats {
    port => 5044
    host => "localhost"
  }
}
```

Ilustración 7- Configuración LogStash input

En el apartado output se configura cual será el destino de la información que ha refinado LogStash. En este caso se ha utilizado como destino ElasticSearch en la ip *localhost:9200* y cuál es el formato de los nombres de los ficheros donde se recoge la información refinada de cada log.

Finalmente se especifica la salida por la salida estándar a través del plugin *rubydebug*, es decir, por pantalla para poder visualizar los datos por consola a modo de depuración.

```
output {
    elasticsearch {
        index => "logauth-%{+YYYY.MM.dd}"
        hosts => ["localhost:9200"]
    }
    stdout { codec => rubydebug }
}
```

Ilustración 8- Configuración de LogStash output

Definidas la salida y entrada de la información, queda definir cuál será el refinamiento de la información. Para ello se ha hecho uso del plugin Grok con la siguiente configuración, que también puede localizarse en el anexo de forma más detallada:

```
    grok {
      match => { "message" => ["%{SYSLOGTIMESTAMP:[system][auth][timestamp]} %{SYSLOGHOST:[system][auth][hostname]} sshd(?:\[%{POSINT:[system][auth]
[pid]}\])?: %{DATA:[system][auth][ssh][event]} %{DATA:[system][auth][ssh][method]} for (invalid user )?%{DATA:[system][auth][user]} from %{IPORHOST:[s
ystem][auth][ssh][ip]} port %{NUMBER:[system][auth][ssh][port]} ssh2(: %{GREEDYDATA:[system][auth][ssh][signature]})?",
                "%{SYSLOGTIMESTAMP:[system][auth][timestamp]} %{SYSLOGHOST:[system][auth][hostname]} sshd(?:\[%{POSINT:[system][auth][pid]}\])?: %{D
ATA:[system][auth][ssh][event]} user %{DATA:[system][auth][user]} from %{IPORHOST:[system][auth][ssh][ip]}",
                "%{SYSLOGTIMESTAMP:[system][auth][timestamp]} %{SYSLOGHOST:[system][auth][hostname]} sshd(?:\[%{POSINT:[system][auth][pid]}\])?: Did
 not receive identification string from %{IPORHOST:[system][auth][ssh][dropped_ip]}",
                "%{SYSLOGTIMESTAMP:[system][auth][timestamp]} %{SYSLOGHOST:[system][auth][hostname]} sudo(?:\[%{POSINT:[system][auth][pid]}\])?: \s*
%{DATA:[system][auth][user]} :( %{DATA:[system][auth][sudo][error]} ;)? TTY=%{DATA:[system][auth][sudo][tty]} ; PWD=%{DATA:[system][auth][sudo][pwd]}
; USER=%{DATA:[system][auth][sudo][user]} ; COMMAND=%{GREEDYDATA:[system][auth][sudo][command]}",
                "%{SYSLOGTIMESTAMP:[system][auth][timestamp]} %{SYSLOGHOST:[system][auth][hostname]} groupadd(?:\[%{POSINT:[system][auth][pid]}\])?:
 new group: name=%{DATA:system.auth.groupadd.name}, GID=%{NUMBER:system.auth.groupadd.gid}",
                "%{SYSLOGTIMESTAMP:[system][auth][timestamp]} %{SYSLOGHOST:[system][auth][hostname]} useradd(?:\[%{POSINT:[system][auth][pid]}\])?:
new user: name=%{DATA:[system][auth][user][add][name]}, UID=%{NUMBER:[system][auth][user][add][uid]}, GID=%{NUMBER:[system][auth][user][add][gid]}, ho
me=%{DATA:[system][auth][user][add][home]}, shell=%{DATA:[system][auth][user][add][shell]}$",
                "%{SYSLOGTIMESTAMP:[system][auth][timestamp]} %{SYSLOGHOST:[system][auth][hostname]} %{DATA:[system][auth][program]}(?:\[%{POSINT:[s
ystem][auth][pid]}\])?: %{GREEDYMULTILINE:[system][auth][message]}"] }
      pattern_definitions => {
        "GREEDYMULTILINE"=> "(.|\n)*"
      }
      remove_field => "message"
    }
```

Ilustración 9- Configuración del filtro Grok de LogStash

Grok se utiliza para depurar y filtrar la información que se encuentra almacenada sin poder recuperarse de una forma sencilla.

```
  date {
    match => [ "[system][auth][timestamp]", "MMM  d HH:mm:ss", "MMM dd HH:mm:ss" ]
  }
  geoip {
    source => "src_ip"
    target => "geoip"
    add_field => [ "[geoip][coordinates]", "%{[geoip][longitude]}" ]
    add_field => [ "[geoip][coordinates]", "%{[geoip][latitude]}"  ]
}
mutate {
    convert => [ "[geoip][coordinates]", "float"]
}
```

Ilustración 10- Filtro final de LosStash

Con este filtro se consigue obtener la fecha en la que se produjo el evento almacenado en el log, y finalmente se hace uso del plugin geoip para obtener las coordenadas de la dirección IP a través de su geolocalización. Esto permitirá más adelante poder segmentar el origen de las conexiones.

4.4.Despliegue de los servicios

El despliegue de los servicios debe de realizarse de la siguiente forma y orden, para que se realice de forma adecuada. Consta de dos fases carga de datos y visualización de datos.

4.4.1. Despliegue de los servicios, carga de datos

Esta fase consiste en el proceso en el que nuevos Logs son cargados a la arquitectura ELK.

ElasticSearch

Primero debe de desplegarse Elastic Search, ya que no necesita del resto de servicios para funcionar y puede quedarse esperando a recibir conexiones entrantes.

systemctl start elasticsearch

Con este comando se está indicando a que el servicio Elastic Search debe de iniciarse, para ello se ha utilizado el comando del sistema *systemctl*.

Para comprobar que el servicio se encuentra activo para recibir datos de LogStash

systemctl status elasticsearch

```
[0x36: /home/hex]# systemctl status elasticsearch
● elasticsearch.service - Elasticsearch
   Loaded: loaded (/usr/lib/systemd/system/elasticsearch.service; disabled; vend
   Active: active (running) since Sat 2018-09-15 17:54:31 CEST; 1 day 4h ago
     Docs: http://www.elastic.co
 Main PID: 7816 (java)
    Tasks: 212 (limit: 4915)
   CGroup: /system.slice/elasticsearch.service
           ├─7816 /usr/bin/java -Xms1g -Xmx1g -XX:+UseConcMarkSweepGC -XX:CMSIni
           └─7884 /usr/share/elasticsearch/modules/x-pack/x-pack-ml/platform/lin

sep 15 17:54:31 0x36 systemd[1]: Started Elasticsearch.
```

Ilustración 11- Comprobación del estado de Elastic Search

Una vez está arrancado se puede comprobar si está listo para recibir peticiones mediante una llamada desde el comando curl a la dirección IP localhost y puerto 9200

curl localhost:9200

```
[0x36: /home/hex]# curl localhost:9200
{
  "name" : "Jv0rVR-",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "xJ2wXLGlR1Wr3Ki7h891Dw",
  "version" : {
    "number" : "6.3.2",
    "build_flavor" : "default",
    "build_type" : "deb",
    "build_hash" : "053779d",
    "build_date" : "2018-07-20T05:20:23.451332Z",
    "build_snapshot" : false,
    "lucene_version" : "7.3.1",
    "minimum_wire_compatibility_version" : "5.6.0",
    "minimum_index_compatibility_version" : "5.0.0"
  },
  "tagline" : "You Know, for Search"
}
```

Ilustración 12- Comprobación de la accesibilidad de Elastic Search

LogStash

El siguiente servicio a desplegar es LogStash que se encarga de recibir la información sin formato y convertir la información a partes más pequeñas para que puedan ser almacenadas en Elastic Search. Por lo que se trata de otro servicio que puede desplegarse a la espera de recibir información que después procesará y enviará a Elastic Search.

/usr/share/logstash/bin/logstash -f /etc/filebeat/pipeline_custom.conf --config.reload.automatic --path.settings=/etc/logstash

Se deberá tener una salida por pantalla del siguiente tipo:

```
[0x36: /etc/logstash]# /usr/share/logstash/bin/logstash -f /etc/filebeat/pipeline_custom.conf --config.reload.automatic --path.settings
=/etc/logstash
Sending Logstash's logs to /var/log/logstash which is now configured via log4j2.properties
[2018-09-07T12:47:15,162][WARN ][logstash.config.source.multilocal] Ignoring the 'pipelines.yml' file because modules or command line o
ptions are specified
[2018-09-07T12:47:15,943][INFO ][logstash.runner          ] Starting Logstash {"logstash.version"=>"6.3.2"}
[2018-09-07T12:47:24,242][INFO ][logstash.pipeline        ] Starting pipeline {:pipeline_id=>"main", "pipeline.workers"=>4, "pipeline.b
atch.size"=>125, "pipeline.batch.delay"=>50}
[2018-09-07T12:47:24,875][INFO ][logstash.filters.geoip   ] Using geoip database {:path=>"/usr/share/logstash/vendor/bundle/jruby/2.3.0
/gems/logstash-filter-geoip-5.0.3-java/vendor/GeoLite2-City.mmdb"}
[2018-09-07T12:47:25,444][INFO ][logstash.inputs.beats    ] Beats inputs: Starting input listener {:address=>"0.0.0.0:5044"}
[2018-09-07T12:47:25,478][INFO ][logstash.pipeline        ] Pipeline started successfully {:pipeline_id=>"main", :thread=>"#<Thread:0x7
4164875 run>"}
[2018-09-07T12:47:25,655][INFO ][org.logstash.beats.Server] Starting server on port: 5044
[2018-09-07T12:47:25,672][INFO ][logstash.agent           ] Pipelines running {:count=>1, :running_pipelines=>[:main], :non_running_pip
elines=>[]}
[2018-09-07T12:47:26,048][INFO ][logstash.agent           ] Successfully started Logstash API endpoint {:port=>9600}
```

Ilustración 13- LogStash iniciado correctamente

Comprobación de que el servicio se encuentra activo:

ps –aux | grep logstash

Se debe de recibir información con el proceso arrancado.

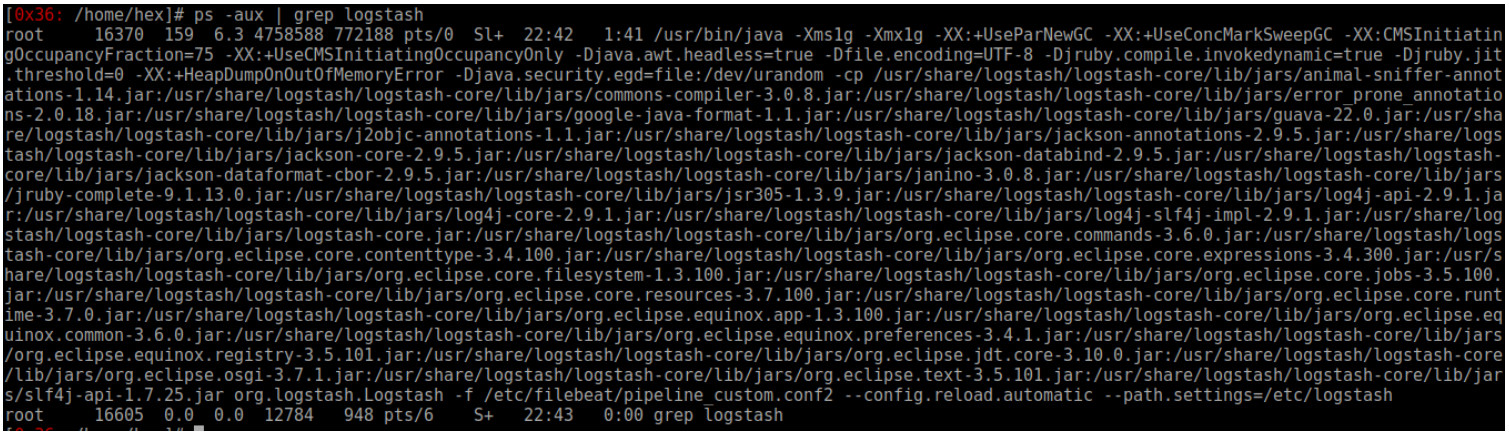

Ilustración 14- Información de ejecución sobre LogStash

Filebeat

El siguiente servicio a desplegar es LogStash que se encarga de recibir la información sin formato y convertir la información a partes más pequeñas para que puedan ser almacenadas en Elastic Search. Por lo que se trata de otro servicio que puede desplegarse a la espera de recibir información que después procesará y enviará a Elastic Search.

/usr/share/filebeat/bin/filebeat -e -c /etc/filebeat/filebeatPROCESS_AUTH_ALL.yml -d "publish"

Comprobación de que el servicio se encuentra activo:

ps –aux | grep filebeat

Se debe de recibir información con el proceso arrancado.

Una vez se tienen los 3 servicios arrancados se produce una conexión entre ambos, que les permite funcionar:

Se debe de visualizar como viaja la información de un servicio a otro que se trata de la salida del plugin de Ruby, que imprime la información por pantalla.

Cuando ha finalizado la carga de datos los servicios de LogStash y FileBeat no son necesarios por lo que se puede acabar con ellos mediante la señal de interrupción CRTL+C en la consola donde se están ejecutando o bien matando el proceso encargado de cada servicio. El proceso encargado de cada servicio puede consultarse como se realizó anteriormente:

ps –aux | grep filebeat

y/ó

ps –aux | grep logstash

Cuándo se tiene el PID se puede acabar con el proceso con el comando kill, siendo de la siguiente forma:

kill -9 PID_LOGSTASH_O_FILEBEAT

Dónde el *PID_LOGSTASH_O_FILEBEAT* se corresponde con el PID consultado si procede.

4.4.2. Despliegue de los servicios, visualización de los datos almacenados

Para poder visualizar los datos se precisa de desplegar el servicio Kibana y tener activo ElasticSearch, pues se trata del elemento encargado de ofrecer una interfaz web que permita visualizar los datos que se han almacenado en Elastic Search.

Para iniciar el servicio Kibana

systemctl start kibana

Con esto se debería de haber iniciado el servicio Kibana, se puede comprobar el estado del servicio mediante:

systemctl status kibana

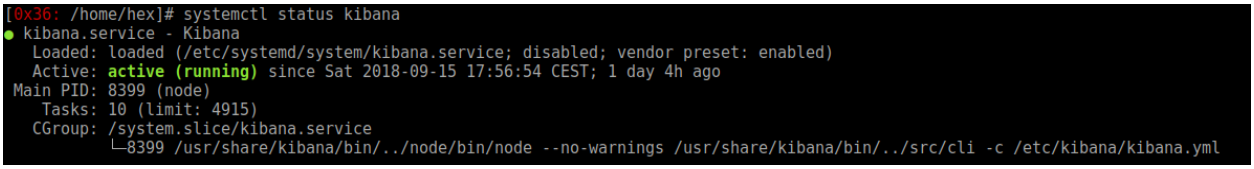

Ilustración 15- Comprobación del estado de Kibana

Se puede comprobar además que se puede acceder al servicio mediante el comando curl, que devolverá el código html ofrecido por Kibana o bien un error por consola:

curl localhost:5601

Si todo se ha desplegado correctamente se podrá acceder a la interfaz web de Kibana, mediante el navegador web.

Capítulo 5: Análisis de la información

Este capítulo trata del análisis de la información almacenada en ElasticSearch. Por tanto trata de la visualización de datos a través de Kibana, mediante búsquedas en ElasticSearch.

5.1. Acceso a Kibana

Por tanto la primera tarea que se realiza el acceder a Kibana. Se accede mediante un navegador web a la url en la que sirve el acceso web.

http://localhost:5601

5.1.1. Partes clave de Kibana

En este punto se describe las partes de que se han considerado como claves para entender el funcionamiento de Kibana. Si bien no se trata de un manual completo sí que puede encontrarse una guía completa de usuario en la documentación oficial de Kibana [24].

Las partes involucradas en el proceso son:

Index: Se trata de la página principal de Kibana.

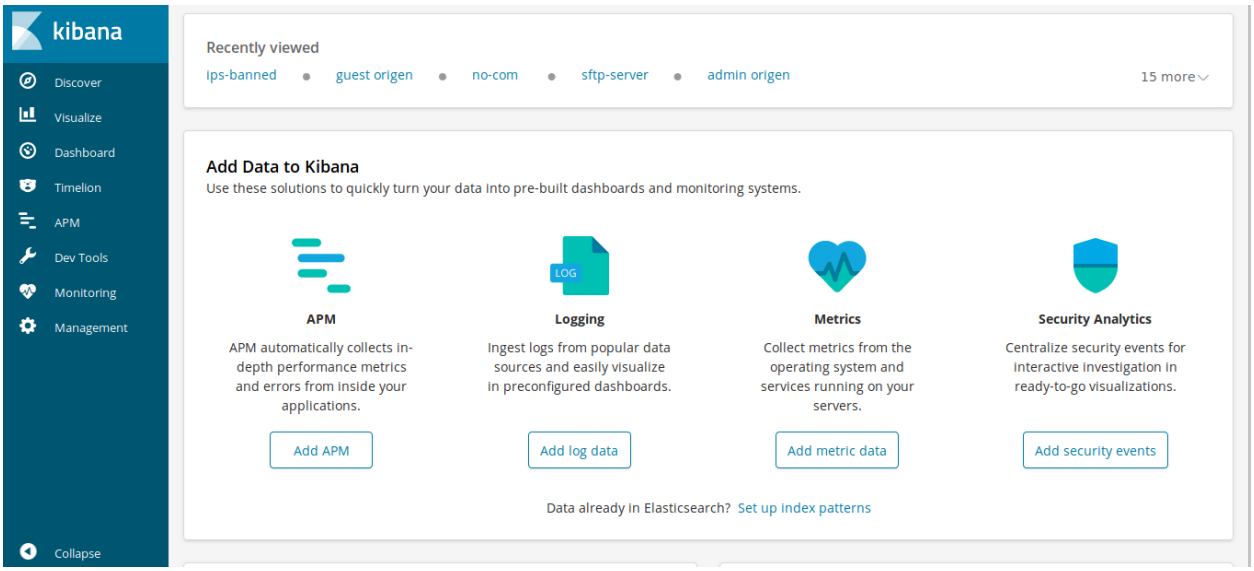

Ilustración 16- Pantalla de inicio de Kibana

Discover: Se trata de la funcionalidad que ofrece Kibana para consultar los datos que se encuentran en Kibana. Permite realizar consultas y guardar los datos para recuperarlos más tarde. Sirve para probar consultas y probar configuraciones.

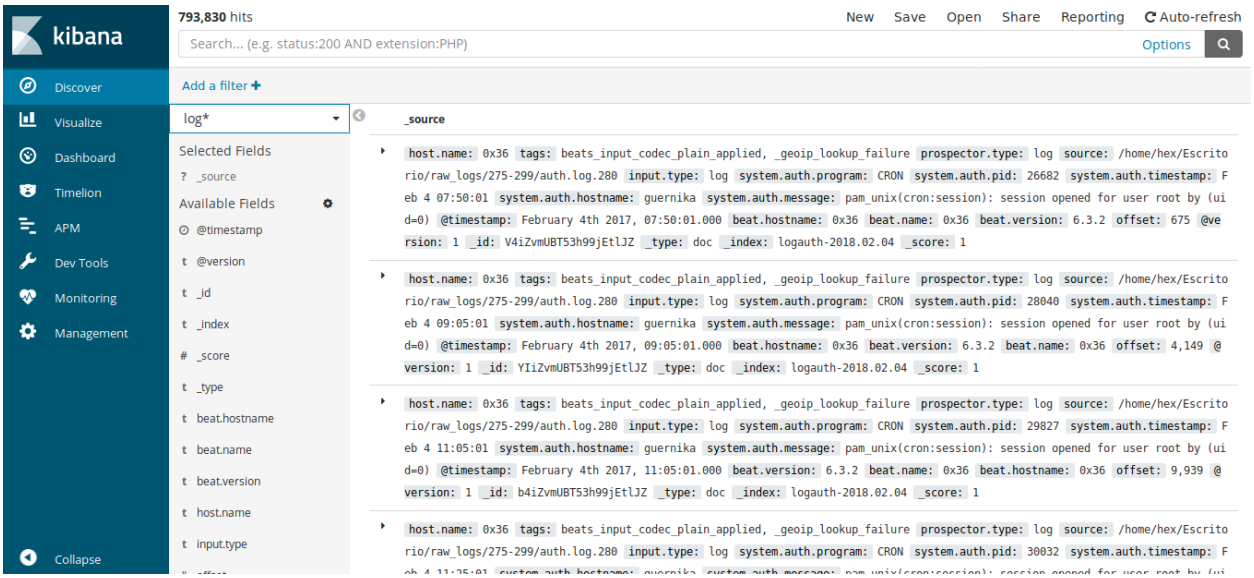

Ilustración 17- Pantalla de discover en Kibana

Visualize: Permite configurar distintos tipos de gráficos en base a consultas realizadas. Existen distintos tipos de gráficos personalizables como gráficos de barras tanto horizontales como verticales, gráficos de tarta, gráficos de líneas etc

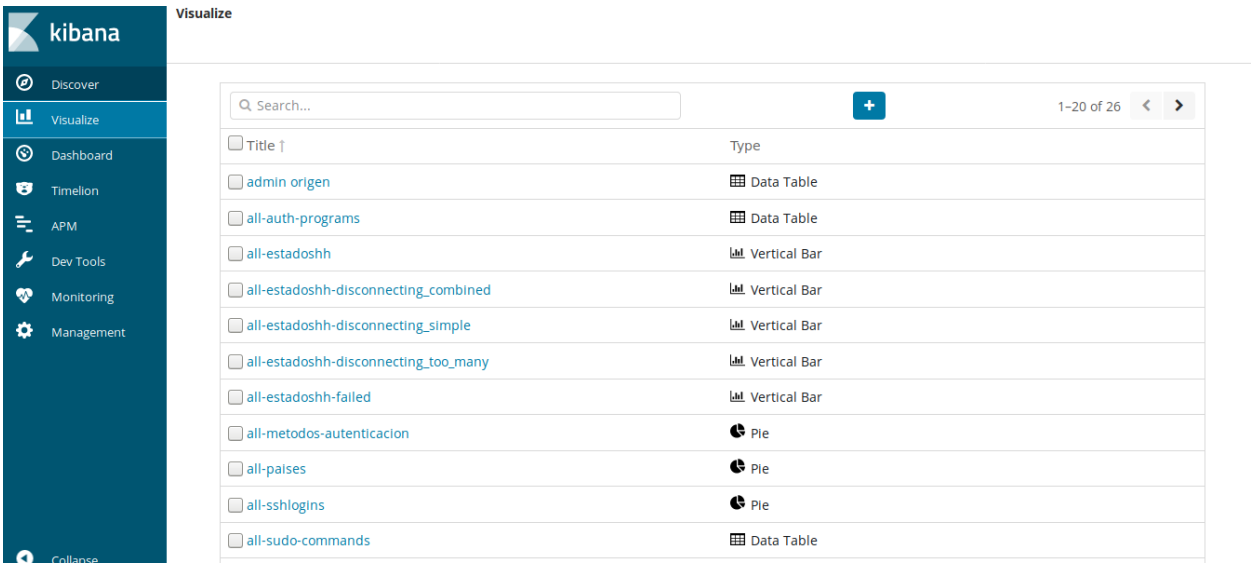

Ilustración 18- Pantalla de visualize en Kibana

Dashboard: Se trata de un panel que permite añadir distintos gráficos creados anteriormente para visualizar varios datos a la vez. Es un panel configurable diseñado para realizar reportes o bien visualizaciones sobre el estado de los datos de forma rápida.

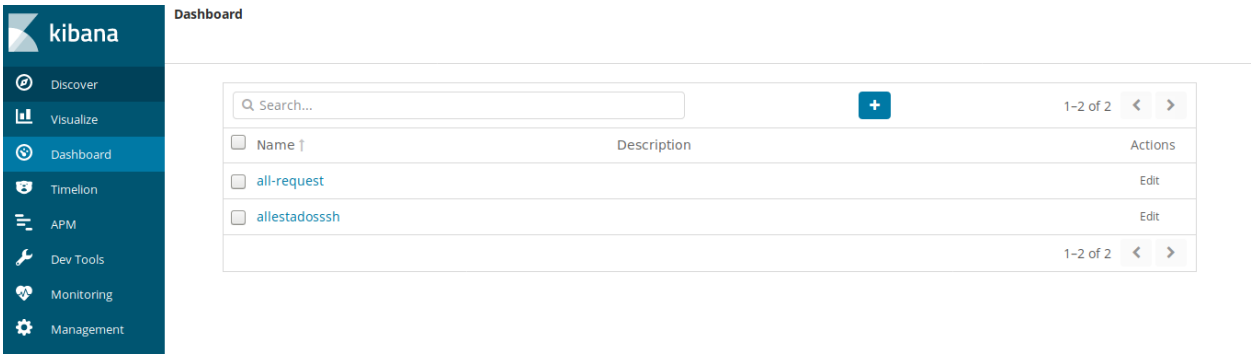

Ilustración 19- Pantalla de Dashboard en Kibana

Management: Se trata de la forma de enlazar los datos de Elastic Search con Kibana. Elastic Search maneja índices de datos, estos índices pueden agruparse para que Kibana pueda consultar la información almacenada en ellos. En el siguiente punto se recoge la configuración realizada.

5.1.2. Configuración de los repositorios a los que accede Kibana

El siguiente paso a realizar es configurar Kibana para leer la información de ElasticSearch pues por defecto esta información no está configurada.

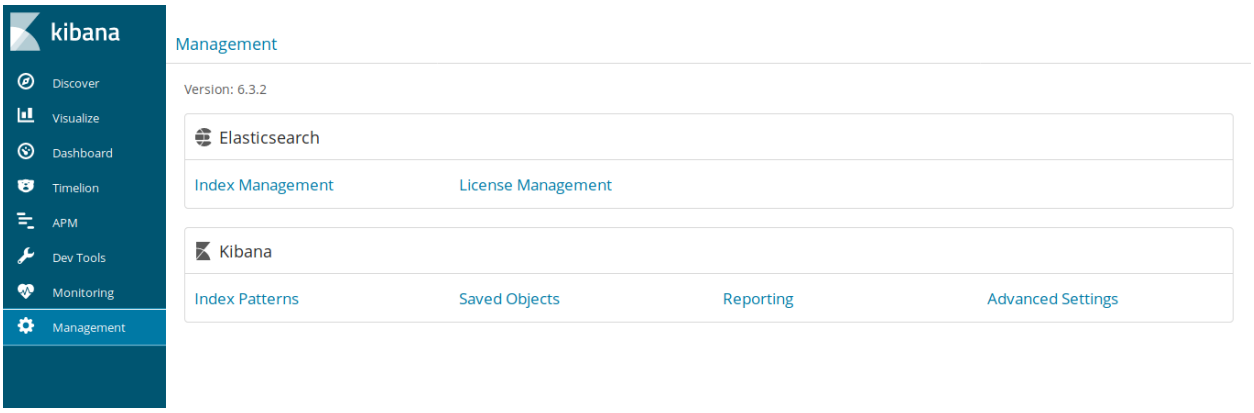

Ilustración 20- Pantalla de Management en Kibana

Seleccionado Index-Patterns se permite al usuario seleccionar y agrupar que logs se quiere visualizar en determinadas consultas que vaya a realizar el usuario a través de los gráficos o de la interfaz de discover.

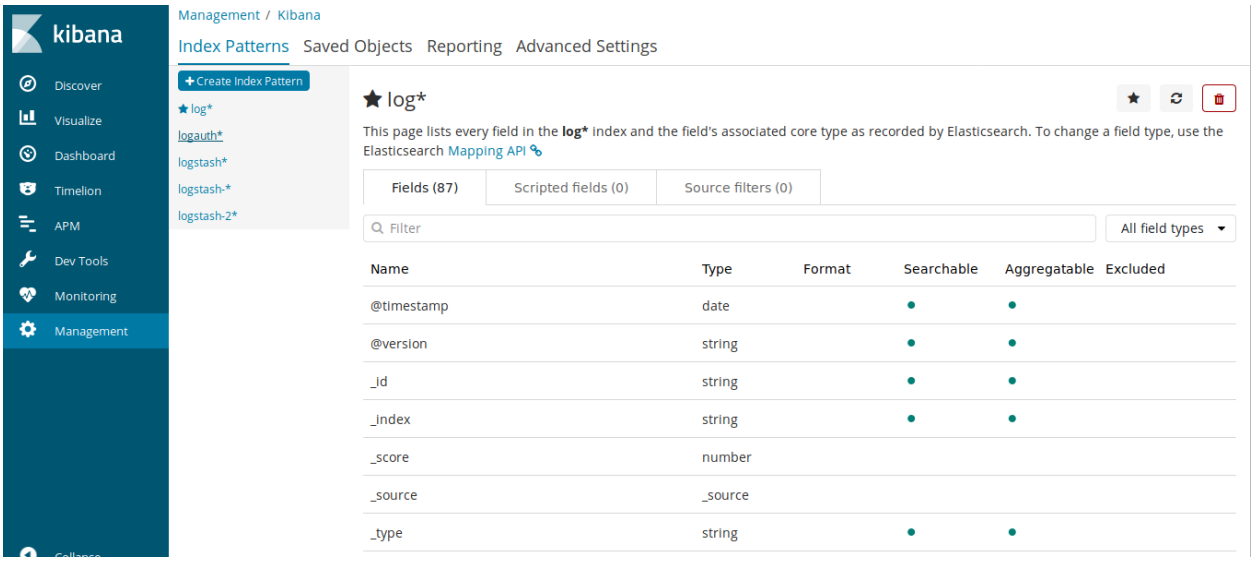

Ilustración 21- Pantalla de configuración Index Patterns en Kibana

Se pueden visualizar los campos de los que están compuestos los datos que se han almacenado en ElasticSearch y crear una nueva agrupación.

En este caso se ha realizado una agrupación a través de *logstash** que se trata del nombre con el que se definieron los logs almacenados en ElasticSearch

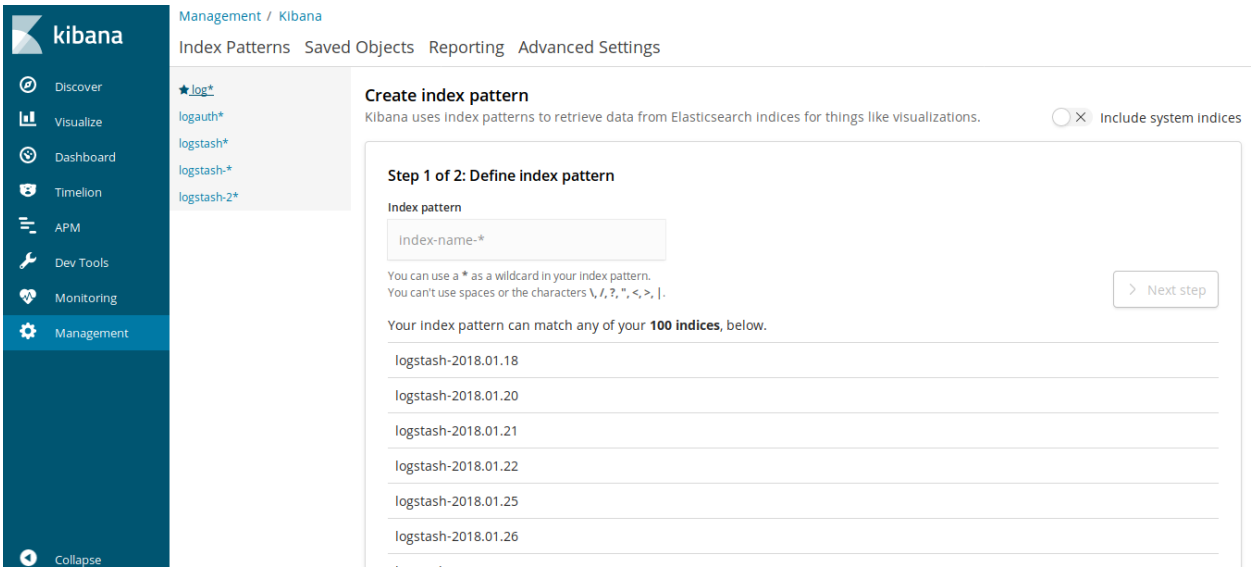

Ilustración 22- Pantalla Index Patterns en Kibana

De esta manera se podrá recuperar la información agrupada de todo el periodo de tiempo en los que se recogió información de Guernika a través de sus logs.

5.1.3. Listado de las consultas utilizadas en Kibana

Programas que escriben en el log

system.auth.program.keyword: Descending

Métodos de autenticación

system.auth.ssh.method.keyword: Descending

Estado de la autenticación

system.auth.ssh.event.keyword: Descending

Estado de la autenticación por días todos los estados

@timestamp per day and system.auth.ssh.event.keywork: Descending

Estado de la autenticación por días filtrado por el estado aceptado

@timestamp per day and system.auth.ssh.event.keywork: “Accepted”

Estado de la autenticación por días filtrado por el estado fallido

@timestamp per day and system.auth.ssh.event.keywork: “Failed”

Estado de la autenticación por días filtrado por el estado invalido

@timestamp per day and system.auth.ssh.event.keywork: “Invalid”

Estado de la autenticación por días filtrado por el estado Disconnecting

@timestamp per day and system.auth.ssh.event.keywork: “Disconnecting”

Estado de la autenticación por días filtrado por el estado too many connections

@timestamp per day and system.auth.ssh.event.keywork: “Disconnecting: Too many authentication failures for invalid”

Geolocalización por países

system.auth.ssh.geoip.country_name.keyword: Descending

Geolocalización por continente

system.auth.ssh.geoip.continent_name.keyword: Descending

Geolocalización por ciudad

system.auth.ssh.geoip.city_name.keyword: Descending

Comandos utilizados con permisos de superusuario

system.auth.sudo.command.keyword: Descending

Usuarios que no tienen permisos de superusuario

system.auth.sudo.error.keyword: “user NOT in sudoers” and system.auth.user.keyword: Descending

Rutas desde las que se han tratado de ejecutar comandos con permisos de superusuario

System.auth.sudo.pwd.keyword: Descending

Comandos fallidos como superusuario por usuario

system.auth.sudo.error.keyword: “user NOT in sudoers” and system.auth.user.keyword: Descending and system.auth.sudo.command.keyword: Descending

Usuarios que a los que se ha intentado cambiar como sudo

system.auth.sudo.user.keyword: Descending

Accesos de root por ciudad, método y resultado:

System.auth.user: root and system.auth.ssh.geoip.city_name.keywork: Descending and system.auth.ssh.method.keywork: Descending and system.auth.ssh.event.keywork: Descending

Accesos de admin por ciudad, método y resultado:

System.auth.user: admin and system.auth.ssh.geoip.city_name.keywork: Descending and system.auth.ssh.method.keywork: Descending and system.auth.ssh.event.keywork: Descending

Accesos de guest por ciudad, método y resultado:

System.auth.user: guest and system.auth.ssh.geoip.city_name.keywork: Descending and system.auth.ssh.method.keywork: Descending and system.auth.ssh.event.keywork: Descending

5.1.4. Configuración de Kibana, dashboard

Esta parte trata de la configuración que se ha realizado para visualizar de distintos gráficos a la vez. Este punto está más orientado a una visualización diaria en la que un administrador visualizaría lo que está pasando con distintas gráficas por lo que al ser datos estáticos de una colección de datos se trata de una foto fija de lo que paso en el servidor Guernika en un periodo de tiempo.

Por lo que este Dashboard se ha utilizado para analizar o ver puntos clave de lo que paso en esa franja de tiempo, quizá sería interesante monitorizar otro tipo de métricas distintas para el día a día pero en el caso evaluado se han utilizado las siguientes como Dashboard.

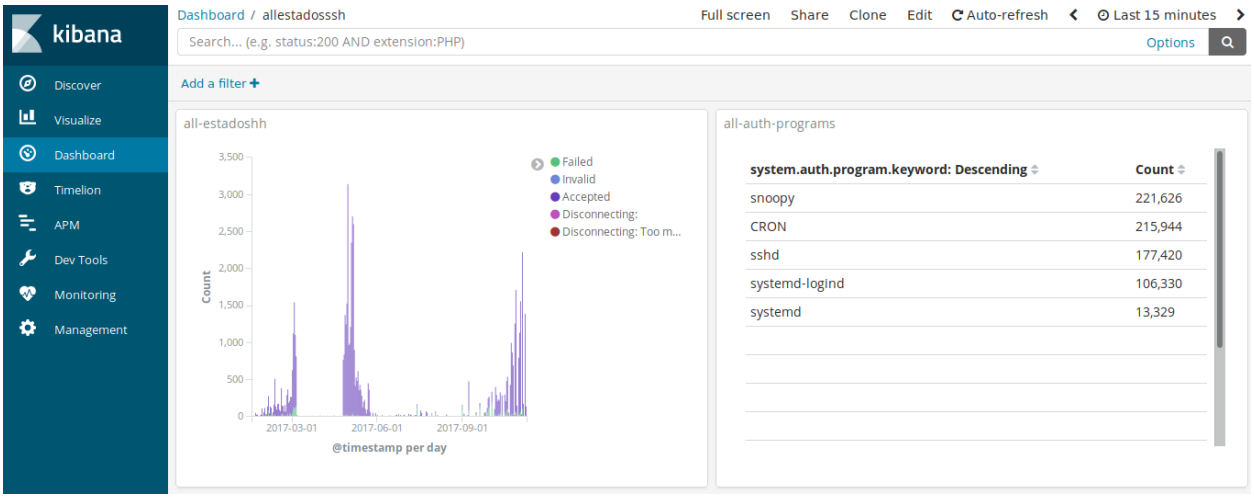

Ilustración 23- Pantalla de Dashboard Kibana

5.2.Hipótesis sobre el análisis

En esta parte del documento se recogen la hipótesis de lo que se espera encontrar durante el análisis de la información.

5.2.1. Hipótesis sobre los tipos de usuario y comportamiento esperados

Este punto recoge los usuarios que se espera encontrar en los logs de Guernika, se especifica tanto el formato del nombre de usuario como el comportamiento que se espera de ellos.

Alumnos y profesores: Estarán identificados por un NIA correspondiente a los alumnos y profesores. Un NIA se trata de un identificador numérico que sigue la siguiente nomenclatura:

100XXXXXX

Donde *XXXXXX* se corresponden con valores numéricos positivos incluyendo el 0. Algunos usuarios pueden disponer de un nombre completo, que se trataría de usuarios creados a medida para profesores o personal interno de la universidad.

Se espera un comportamiento en el que un alumno o profesor se conecta mediante SSH al servidor y ejecuta aplicaciones desarrolladas por el mismo o que le permitan realizar tareas de la universidad a la par de obtener información sobre la máquina de Guernika. Cada alumno o profesor dispone una cuenta en Guernika la cual incluye su propia carpeta personal que se trata de un directorio home.

No se espera que los alumnos dispongan de permisos de root o sudo. Por lo que sólo deberían de disponer de permisos suficientes para manejar los ficheros y directorios fuera de su carpeta personal.

Root: Cuenta administrativa con permisos de súper usuario. Se espera que no tenga ningún tipo de restricción a la hora de realizar tareas.

Usuarios creados para aplicaciones: Es esperable encontrar usuarios creados durante la instalación de un programa con el fin de funcionar correctamente en el equipo con la funcionalidad y permisos que precisen. Un ejemplo de tipos de usuarios podría ser el usuario que utiliza Apache, Cron o MySQL. Se espera que este tipo de usuarios realmente sean utilizados por el sistema y no por usuarios humanos por lo que no se espera un malfuncionamiento o uso inapropiado. Su comportamiento debe restringirse a manejar los datos y directorios de la aplicación a la que pertenece.

5.2.2. Hipótesis sobre la clasificación de comportamientos esperados

Este punto recoge los comportamientos que se espera encontrar por parte de los usuarios en el sistema Guernika.

Normal: Comportamiento que se ajusta al comportamiento esperado para el tipo de cuenta al que pertenece.

Sospechoso: Será un usuario del que se tiene duda acerca de la actividad que ha realizado. Será una alerta.

Atacante: Será un usuario del que se tiene la certeza de haber realizado un comportamiento malicioso. Será una incidencia.

Elementos a los que prestar atención para no confundir con comportamientos maliciosos

- **Errores de programación o programas:** Los errores de programación pueden ocasionar fallos de comportamiento en un sistema por lo que lo que a prori no es un ataque intencionado puede convertirse en un ataque de denegación de servicio. Por lo que buscar un número excesivo de errores de programación o determinados códigos de error en algunas apps puede desvelar fallos de funcionamiento inintencionados.
- **Usuarios torpes o clumsy users:** Usuarios que han configurado de forma incorrecta el sistema o programas. Esto puede ocasionar fallos de funcionamiento inintencionado.

5.2.3. Hipótesis sobre los ataques que se espera encontrar

Guernika al ser un servidor multiusuario abierto a internet se espera que pueda ser objetivo de los siguientes ataques:

- **Denegación de servicio**: Se espera que el servidor en algún momento sea objeto de múltiples conexiones simultáneas, que pueden llevar al servidor a un estado inaccesible o empeore su rendimiento. Un ataque de denegación puede ser intencionado o bien producido por un número inusual de usuarios.
- **Escáner de servicios/puertos**: Se espera que posibles atacantes realicen una fase de reconocimiento en el servidor con el fin de detectar servicios o puertos expuestos a internet.
- **Intentos de inicio de sesión no autorizado**: Se espera que al ser un servidor multiusuario se produzcan intentos de inicio de sesión no autorizados, es decir habrá fallos de credenciales de usuario y contraseña.
- **Ataques de fuerza bruta**: Se espera que se trate de entrar al servidor por medios de fuerza bruta, probando el mismo usuario con distintas contraseñas.
- **Escalada de privilegios**: Se esperar encontrar intentos de utilizar programas con privilegios que no le corresponden.
- **Accesos no autorizados a zonas del equipo**: Se espera encontrar que los usuarios traten de acceder a zonas restringidas para el de usuario que tienen.

- **Programas maliciosos**: Es posible que se descubra que usuarios están ejecutando programas maliciosos en el equipo.
- **Fugas de información**: Se espera encontrar a usuarios accediendo a ficheros que quizá no deberían de acceder.

5.2.4. Hipótesis sobre la detección de un ataque o comportamientos extraños

Para poder detectar un atacante se ha de seguir una serie de pautas. El siguiente esquema que se ha utilizado para detectar un comportamiento de un atacante está basado en el libro *Network Security Through Data Analysis* [25] que define de forma breve las fases de un modelo de ataque.

Fases involucradas en un ataque

Para poder detectar un atacante es preciso identificar las partes que lo componen.

- **Reconocimiento**: El atacante realiza un estudio acerca de su objetivo. Este estudio es distinto en función del objetivo del atacante. Puede consistir desde ingeniería social a un escáner con nmap, donde se obtienen los puertos abiertos y cerrados de una computadora.
- **Subversión**: Consiste en el proceso en el que un atacante aprovecha un exploit y lo utiliza con el fin de obtener un beneficio de una forma no prevista en el sistema. La manera de hacer esto puede variar, puede ser desde un exploit remoto accesible mediante internet, virus, gusanos etc
- **Configuración:** Una vez el atacante dispone de acceso o control del sistema ya sea de forma total o parcial, el atacante modifica o cambia parte del sistema para sus propios fines. Un comportamiento esperado de esta fase puede ser desactivar cortafuegos, habilitar permisos de usuario o instalar software adicional.
- **Exploiting:** Una vez el atacante dispone de control del sistema ahora lo utiliza para un fin en concreto. Este comportamiento es distinto por cada ataque porque las intenciones varían en base al ataque y al objetivo por parte del atacante.
- **Propagación**: Se trata de la fase en la que el atacante dispone de control de una máquina y esta es utilizada para propagar el ataque a otras máquinas con el fin de replicar el ataque. Un ejemplo de esta fase podría ser atacar las computadoras dentro de la red local a través de la máquina comprometida.

Nota: Un ataque puede no cumplir todas las fases pero si parte de ellas, depende del objetivo del atacante.

5.2.5. Hipótesis sobre posibles evidencias de ataques

En este punto se recogen aspectos a visualizar y monitorizar dentro de los resultados para detectar evidencias de ataques o accesos no autorizados al equipo.

Intentos de inicio de sesión SSH

Aunque puede que no se trate de un ataque, un número inusualmente alto de intentos de inicio de sesión puede indicar que una cuenta se encuentra comprometida o se está intentando suplantar la identidad de un usuario. También en el peor de los casos puede indicar un ataque de fuerza bruta al servidor SSH.

Intentos de inicio de sesión SSH, frecuencia

Si se obtiene que un número inusual de intentos de inicio de sesión se han producido en un espacio de tiempo muy pequeño se puede considerar como un intento de denegación de servicio.

Geolocalización de los usuarios

Debido a que se dispone de la información de conexión de cada usuario, en concreto la dirección IP se ha procedido a realizar la técnica de *reverse DNS resolution (RDNS).*

La técnica RDNS consiste en consultar una dirección IP en el sistema de nombres de dominio de internet para obtener el nombre de dominio. Es el proceso inverso a obtener una dirección IP a través de un nombre de dominio.

En concreto esta técnica se va a realizar para geolocalizar el conjunto de direcciones IP que se han obtenido a través de la información disponible mediante el login SSH, pues se conoce la dirección IP de donde proviene el login frente al servidor Guernika.

Para realizar esta tarea se utiliza la librería GeoIp. Se trata de un paquete instalable en la mayoría de distribuciones Linux que permite a través de una dirección IP geolocalizar la conexión desde donde proviene. Si bien no se trata de una geolocalización exacta si provee de forma aproximada el origen de la conexión, pudiendo trazar y clasificar las conexiones por países.

El objetivo de geolocalizar las conexiones que ha recibido Guernika por parte de los usuarios es detectar patrones de comportamiento extraños.

Comportamientos considerados como peligrosos

- Lectura de directorios sobre los que no se tiene permiso.
- Intento de uso de root o sudo.
- Escalada de privilegios.
- Un número inusual de peticiones a otras máquinas dentro de la misma red local.
- Modificación del sistema operativo.

Número inusual de peticiones ip a la red local por parte de Guernika

Un número inusual de peticiones a máquinas dentro de la misma red local puede indicar algún tipo de infección que trata de propagarse hacia los equipos de la misma red.

5.2.6. Otros datos a tener en cuenta

Comandos más utilizados

Mediante el conteo de los comandos utilizados por todos los usuarios en el equipo en todas las sesiones se puede determinar cuáles son los comandos más utilizados y por tanto relacionarlos con patrones de comportamiento comunes.

Comandos menos utilizados o utilizados una sola vez

Realizar un análisis de lo menos común puede revelar intenciones y comportamientos que se desvían de lo esperable.

Horas de conexión más frecuentes

Se dispone de la fecha y hora de la conexión. Por lo que se puede utilizar para determinar las horas más frecuentes de conexión y detectar patrones de conexiones anormales como podría ser un número inusual de conexiones en un determinado momento que podría indicar algún tipo de ataque de denegación de servicio.

Horas de conexión menos frecuente de actividad

Al igual que se pueden detectar picos de servicio de servicio inusuales por encima de un día de actividad normal, también es posible detectar la inversa. Es decir un número de peticiones inusualmente bajo puede indicar problemas de funcionamiento o cortes de servicio.

5.3. Problemas en el análisis

Como se adelantaba en el punto *Conclusiones acerca del marco regulador, capítulo 1*. Para poder publicar el estudio sin caer en una infracción legal es necesario utilizar usuarios anónimos que no identifiquen de ninguna forma a una persona real.

El problema es que se realizó la carga de datos al sistema de ElasticSearch sin tener en cuenta esta consideración. Por lo que se ha tenido que realizar la ofuscación de los datos a posteriori para solventar el problema. Para solventar el problema se ha decido utilizar como método de cifrado el método OneTimePad.

5.3.1. Ofuscación de los datos, One Time Pad

El método de cifrado reversible estará basado en el método One-Time-Pad. Este método se trata de un método de cifrado fiable siempre y cuando se cumplan las siguientes premisas:

1. La clave ha de ser de la misma longitud que la cadena que se va a cifrar, en este caso el nombre de usuario.
2. La clave debe de ser aleatoria.
3. No se debe de repetir la clave en ningún mensaje.

Por lo que el problema principal es la forma de generar una clave totalmente aleatoria. Para ello se ha utilizado el *ANU Quantum Random Number Server*.

Se trata de un generador cuántico de números aleatorios alojado en el departamento de computación cuántica de la universidad Australian National University en Acton, Camberra, Australia. [26] [27].

Este generador cuántico de números aleatorios está basado en las mediciones de las fluctuaciones cuánticas del vacío. El laboratorio ofrece una API para que cualquiera pueda realizar una consulta a este generador. Para realizar esta tarea se ha desarrollado un script en Python con el siguiente diseño:

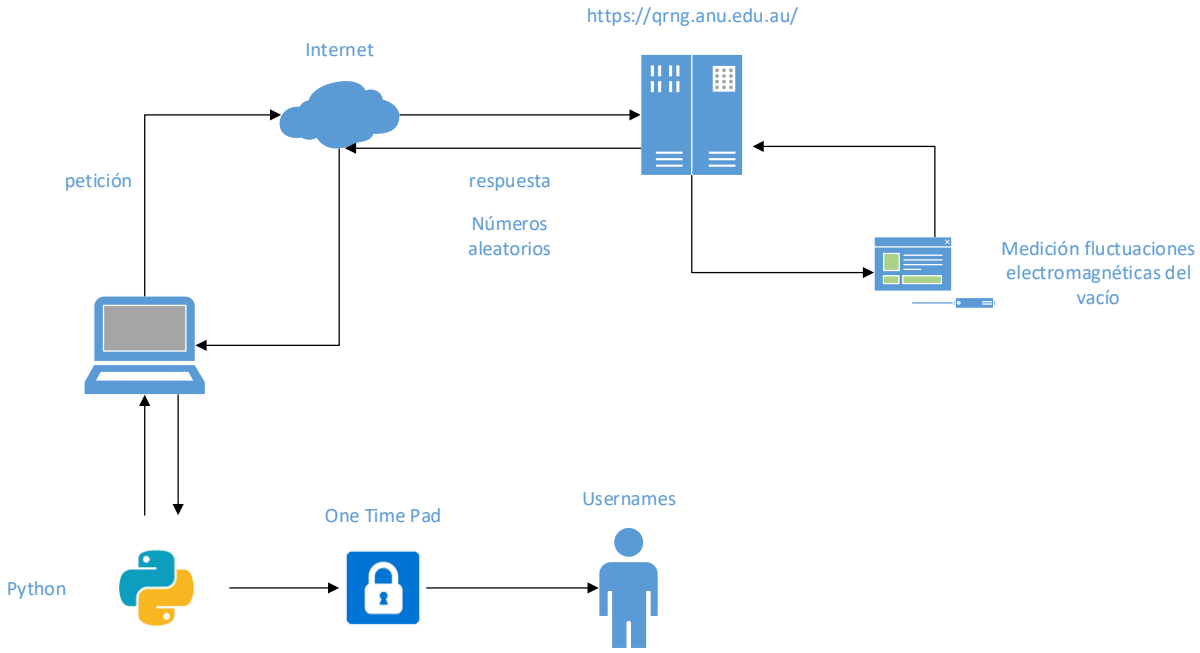

Ilustración 24- Esquema de diseño del ofuscador de datos

Su funcionamiento se basa en lo siguiente:

1. Lee los usuarios en formato *CSV* separados por comas. Este fichero se puede obtener de Kibana cuando se realiza una consulta por el campo *system.auth.usr.keyword.*
2. Obtiene el número de usuarios que se han localizado en el sistema por medio del formato CSV.
3. Solicita la misma cantidad de números aleatorios en formato hexadecimal al generador de números aleatorios al servidor [28].
4. Utiliza un número aleatorio distinto en formato hexadecimal para cifrar cada nombre de usuario.
5. Muestra por la salida estándar el cambio realizado.

5.4.Estudio de los logs

Este punto recoge el estudio que se ha realizado con la información refinada que se ha almacenado en Elastic Stack.

5.4.1. Periodo de tiempo evaluado

El periodo de tiempo evaluado a través de los logs ocurre desde el día 16/1/2017 hasta el día 6/11/2017. Por lo que el tiempo evaluado es de 295 días.

5.4.2. Programas que han escrito en el log

Este punto recoge la información de los distintos programas que han escrito en el Log. Se recoge el total de programas que han recogido información así como el número de veces que lo han hecho.

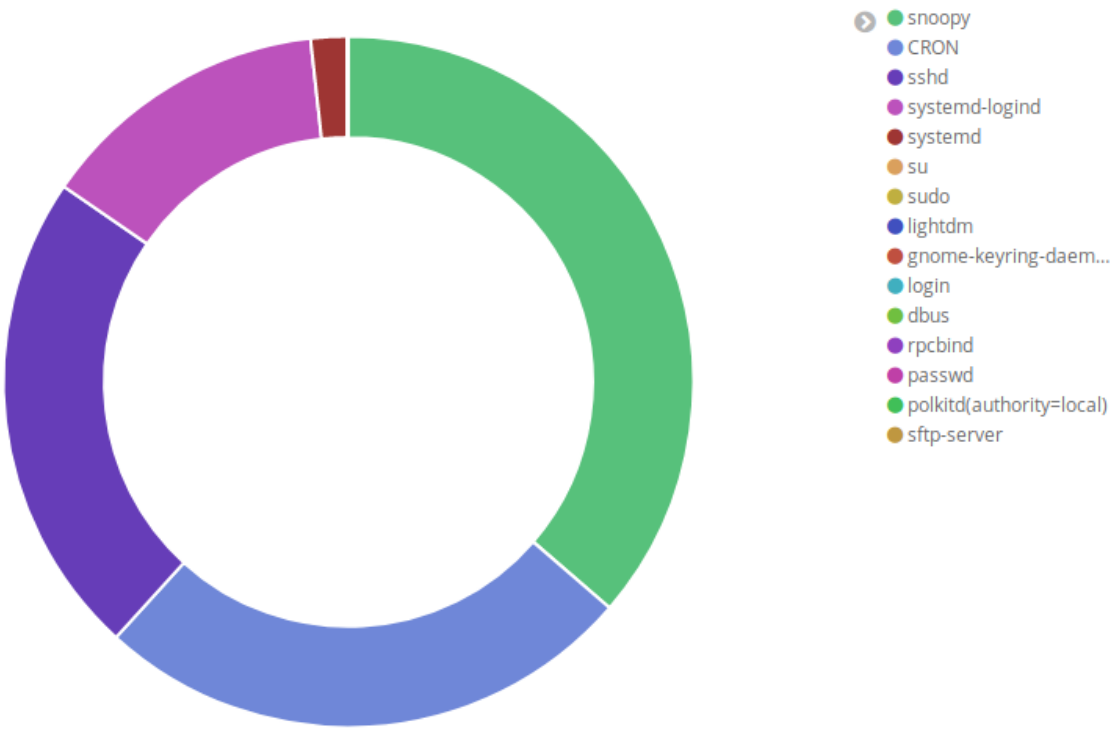

Figura 1- Gráfico correspondiente a los programas que han escrito en el log

La siguiente tabla recoge la información mostrada en el gráfico.

La tabla está compuesta por el programa que escribió en el Log así como el número de veces que lo hizo en el periodo desde el día 2017-02-01 a 2017-11-06.

snoopy	328,915
CRON	230,041
sshd	206,411
systemd-logind	123,814
systemd	15,298
su	224
sudo	154
lightdm	52
gnome-keyring-daemon	44
login	38
dbus	30
rpcbind	17
passwd	12
polkitd(authority=local)	4
sftp-server	2

Tabla 33- Programas que han escrito en log

5.4.3. Método de autenticación

Para poder iniciar sesión en el servidor Guernika se debe de producir una autenticación por parte del servidor hacía el usuario que inicia sesión.

Por lo que uno de los primeros elementos a evaluar es como han iniciado sesión la totalidad de usuarios en el equipo.

Se ha obtenido lo siguiente.

- password: 68,478
- publickey: 1,112
- too many authentication failures: 23
- none: 14

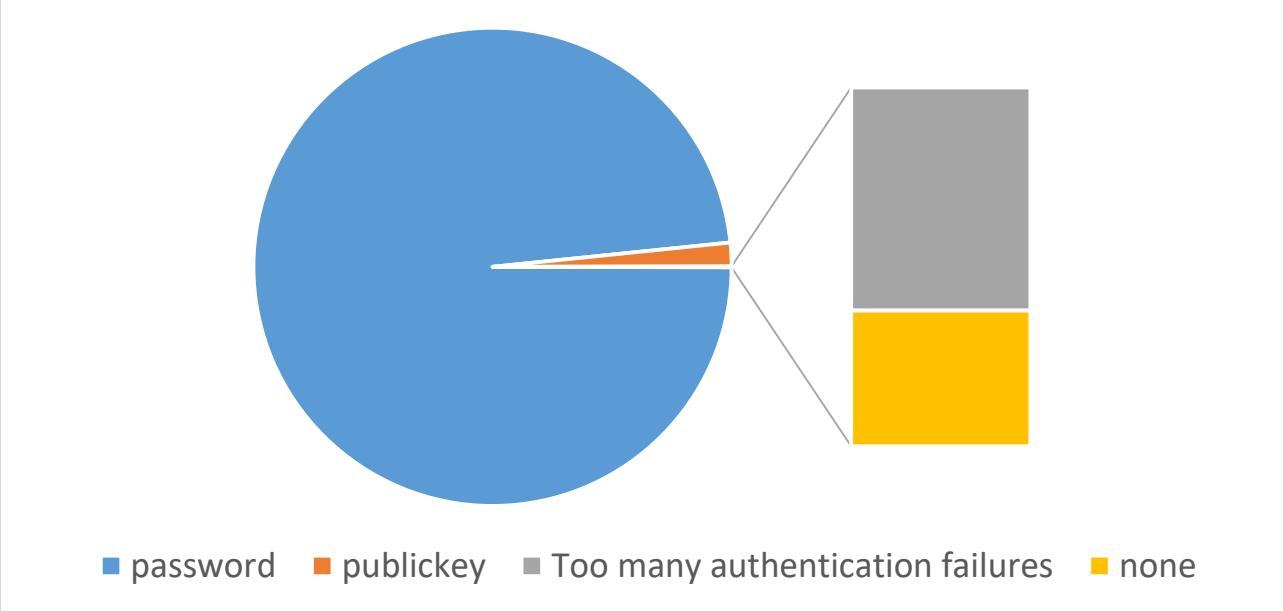

Figura 2- Gráfica sobre los métodos de autenticación utilizados

Obteniendo el peso porcentual sobre el total de autenticaciones realizadas por cada método de autenticación se obtiene lo siguiente:

Método de autenticación	Total	Porcentaje sobre el total
password	68478	98,34978%
publickey	1112	1,597082%
too many authentication failures	23	0,033033%
none	14	0,020107%

Tabla 34- Número de autenticaciones por método

5.4.4. Estado de la autenticación

El siguiente punto evaluado es conocer que ha ocurrido con las autenticaciones. Para ello se ha evaluado tanto las autenticaciones que se han producido con éxito como aquellas que no se han realizado por un fallo de autenticación.

- Accepted: 65,865
- Failed: 3,739
- Invalid: 978
- Disconnecting: 23
- Disconnecting: Too many authentication failures for invalid: 3

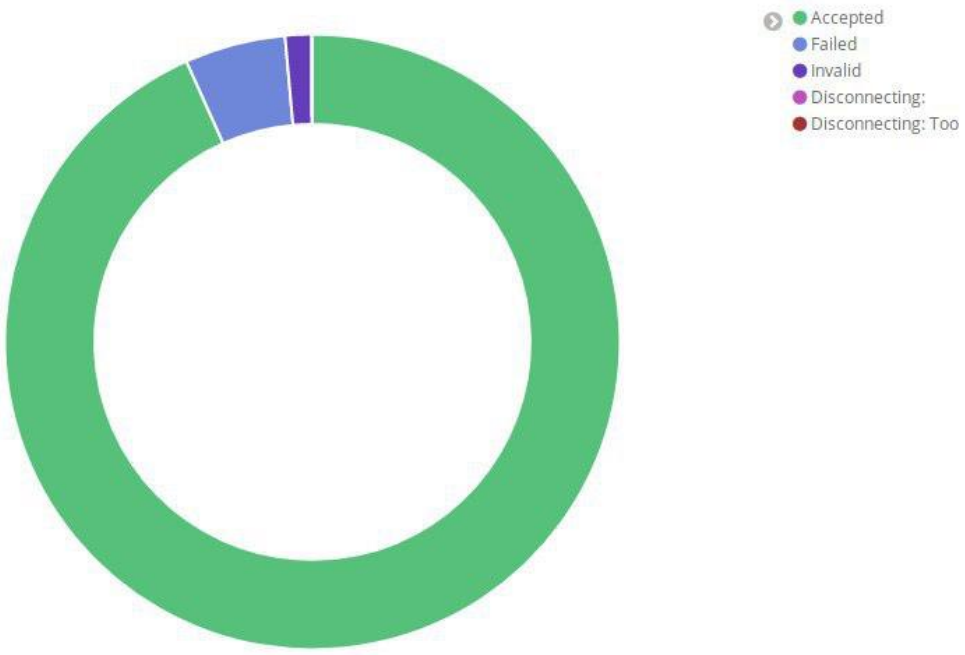

Figura 3- Estado de la autenticación

Método de autenticación	Total	Porcentaje sobre el total
Accepted	65865	93,28263%
Failed	3,739	5,295434%
Invalid	978	1,385112%
Disconnecting	23	0,032574%
Disconnecting: Too many authentication failures for invalid	3	0,004249%

Tabla 35- Total de resultados sobre los métodos de autenticación

5.4.5. Estado de la autenticación por días

Este punto describe como han sido el estado de las conexiones que se han realizado a Guernika durante el periodo de estudio desde la fecha 2017-02-01 a 2017-11-06 evaluado por días. Por tanto el siguiente gráfico recoge en su eje X el periodo de tiempo descrito por días con los distintos estados posibles de la conexión. En el eje Y se recoge el número total de peticiones que se realizaron ese día. La barra vertical, describe en varios colores los estados de las conexiones que se sucedieron en esas fechas. Siendo verde para las fallidas, azul para las inválidas, morado para las conexiones aceptadas, rosa para las desconectadas y finalmente en rojo las desconectadas por múltiples intentos fallidos.

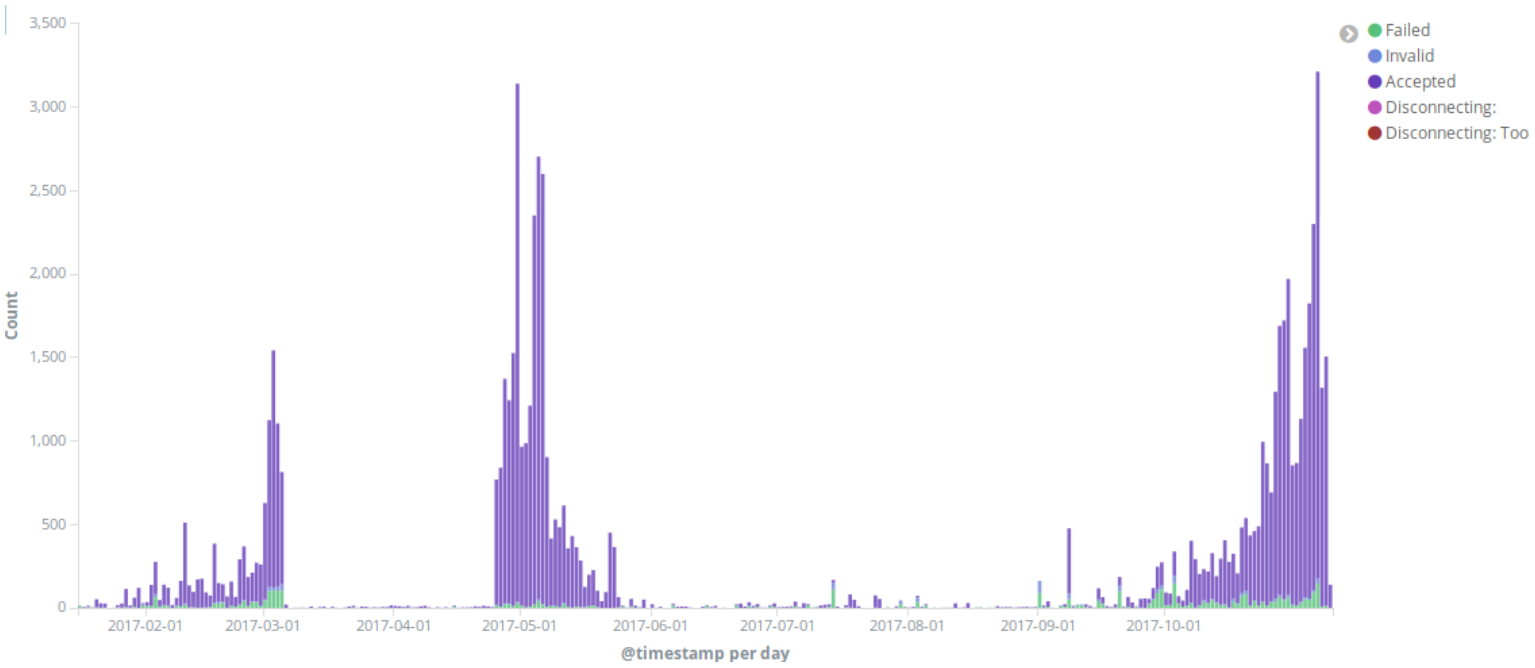

Figura 4- Resultado de la autenticación por dias

5.4.6. Estado de la autenticación filtrado por el estado Failed en días

En este apartado se recoge la gráfica que describe cuales han sido las conexiones cuyo estado ha sido failed filtrado por días. En el eje X por tanto se evalúan los días, desde la fecha 2017-02-01 a 2017-11-06 y en el eje Y se recoge la suma total de apariciones. Cada barra verde representa la relación descrita en el eje X e Y.

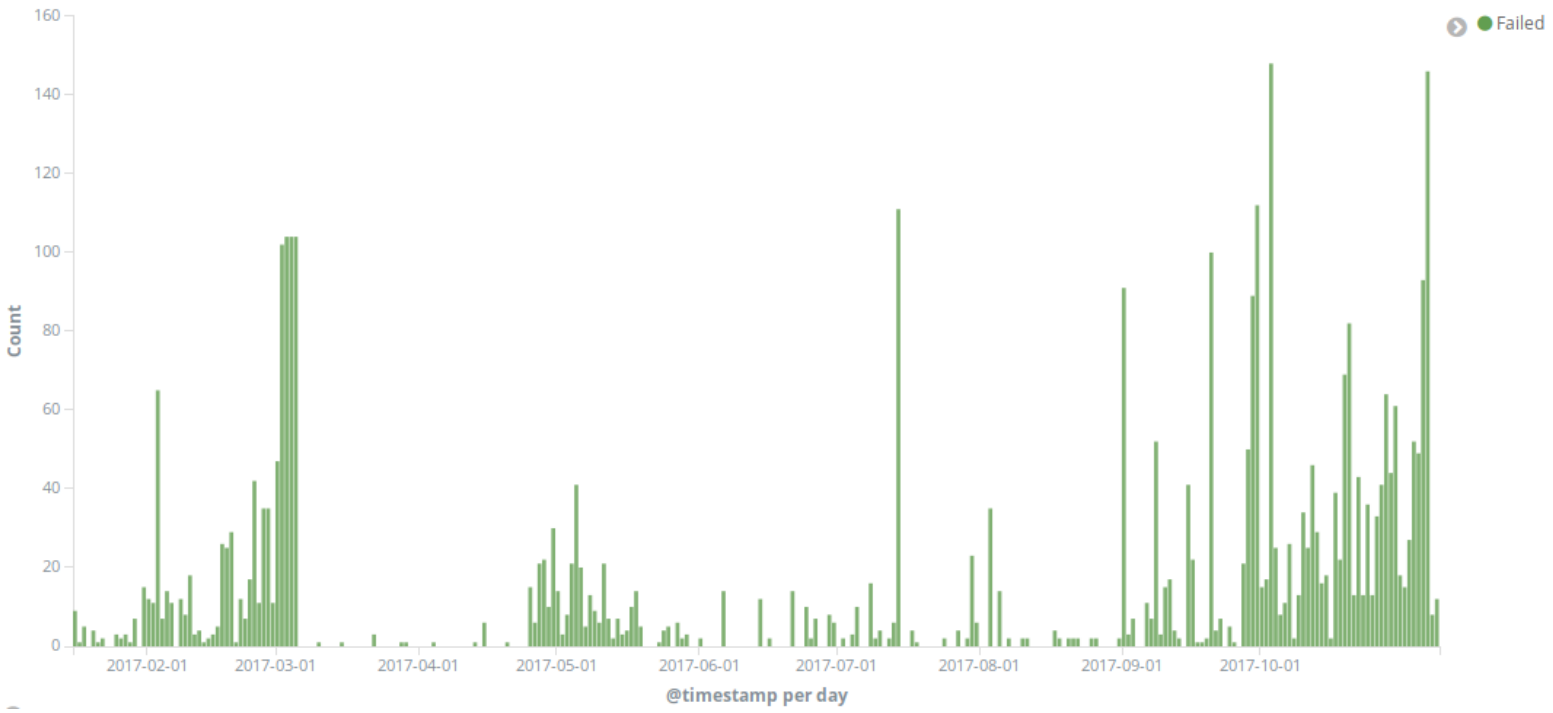

Figura 5- Número de autenticaciones fallidas por día

5.4.7. Estado de la autenticación filtrado por el estado Invalid en días

En este apartado se recoge la gráfica que describe cuales han sido las conexiones cuyo estado ha sido invalid filtrado por días. En el eje X por tanto se evalúan los días, desde la fecha 2017-02-01 a 2017-11-06 y en el eje Y se recoge la suma total de apariciones. Cada barra azul representa la relación descrita en el eje X e Y.

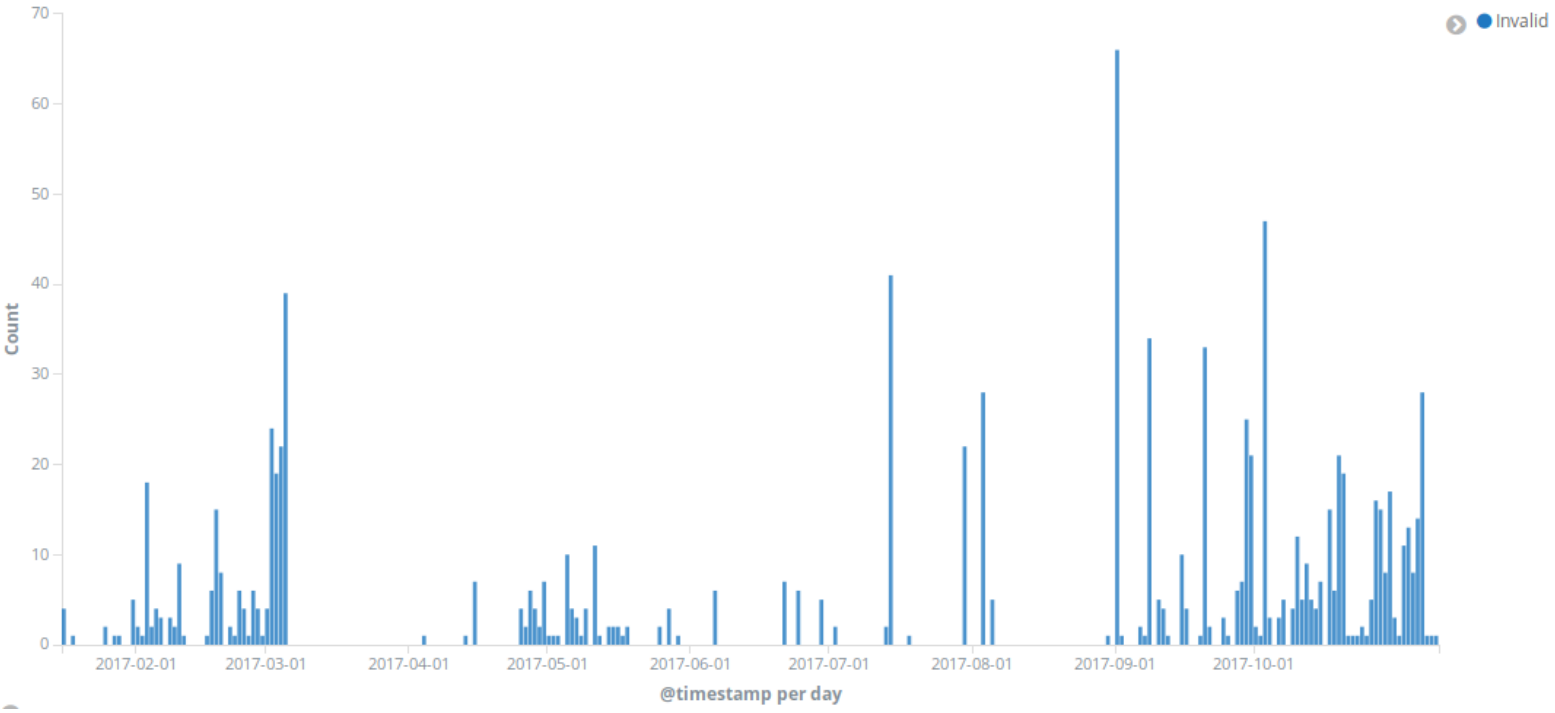

Figura 6- Número de autenticaciones inválidas por día

5.4.8. Estado de la autenticación filtrado por el estado Accepted por días

En este apartado se recoge la gráfica que describe cuales han sido las conexiones cuyo estado ha sido accepted filtrado por días. En el eje X por tanto se evalúan los días, desde la fecha 2017-02-01 a 2017-11-06 y en el eje Y se recoge la suma total de apariciones. Cada barra morada representa la relación descrita en el eje X e Y.

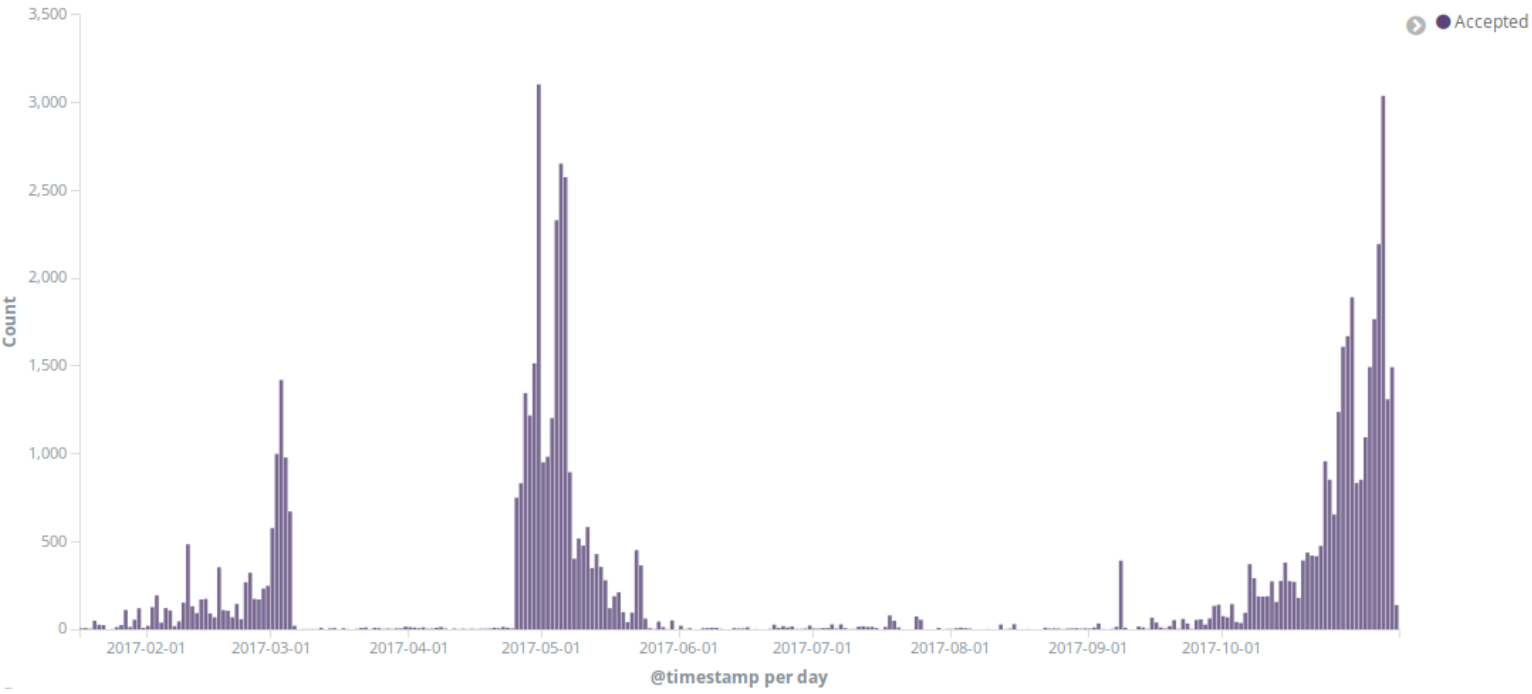

Figura 7- Número de autenticaciones aceptadas por día

5.4.9. Estado de la autenticación filtrado por el estado Disconnecting por días

En este apartado se recoge la gráfica que describe cuales han sido las conexiones cuyo estado ha sido disconnecting filtrado por días. En el eje X por tanto se evalúan los días, desde la fecha 2017-02-01 a 2017-11-06 y en el eje Y se recoge la suma total de apariciones. Cada barra rosa representa la relación descrita en el eje X e Y.

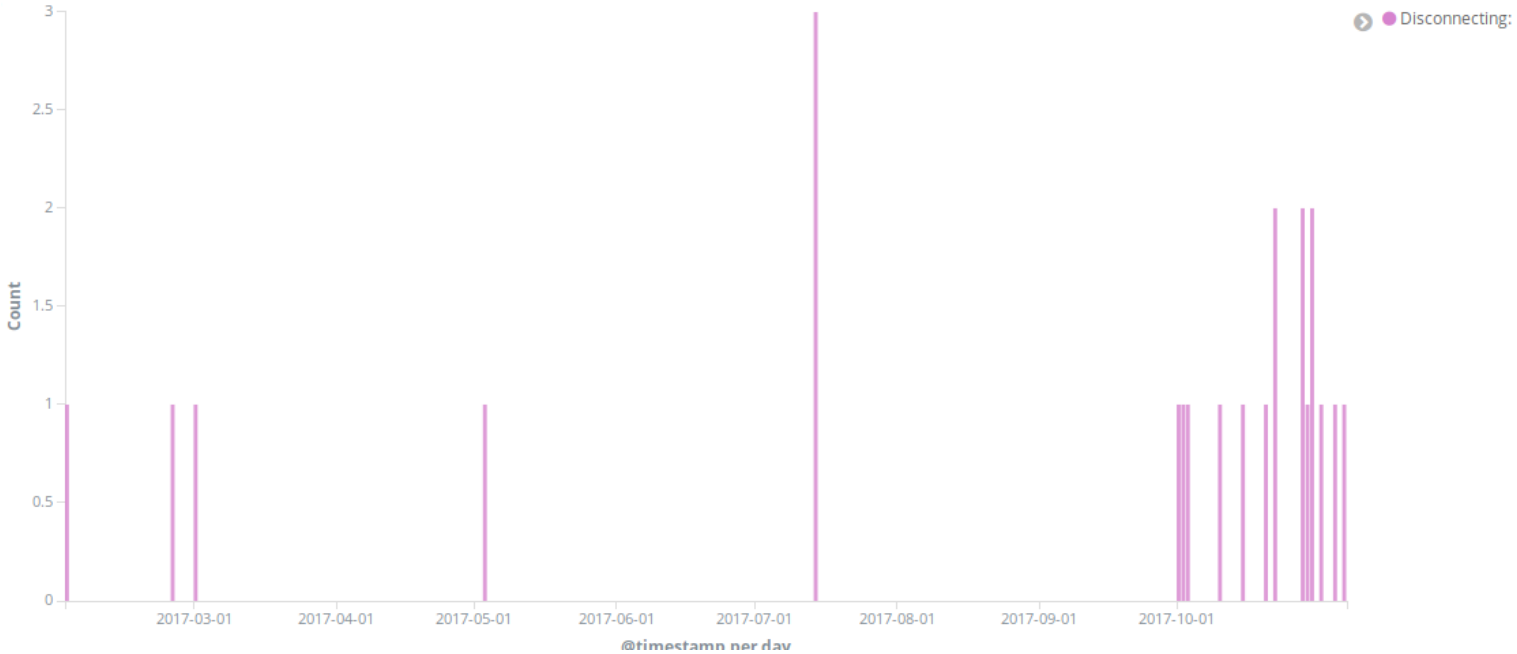

Figura 8- Número de autenticaciones Disconnecting por día

5.4.10. Estado de la autenticación filtrado por el estado Disconnecting: Too many authentication failures for invalid

En este apartado se recoge la gráfica que describe cuales han sido las conexiones cuyo estado ha sido Disconnecting: Too many authentication failures for invalid filtrado por días. En el eje X por tanto se evalúan los días, desde la fecha 2017-02-01 a 2017-11-06 y en el eje Y se recoge la suma total de apariciones. Cada barra morada representa la relación descrita en el eje X e Y.

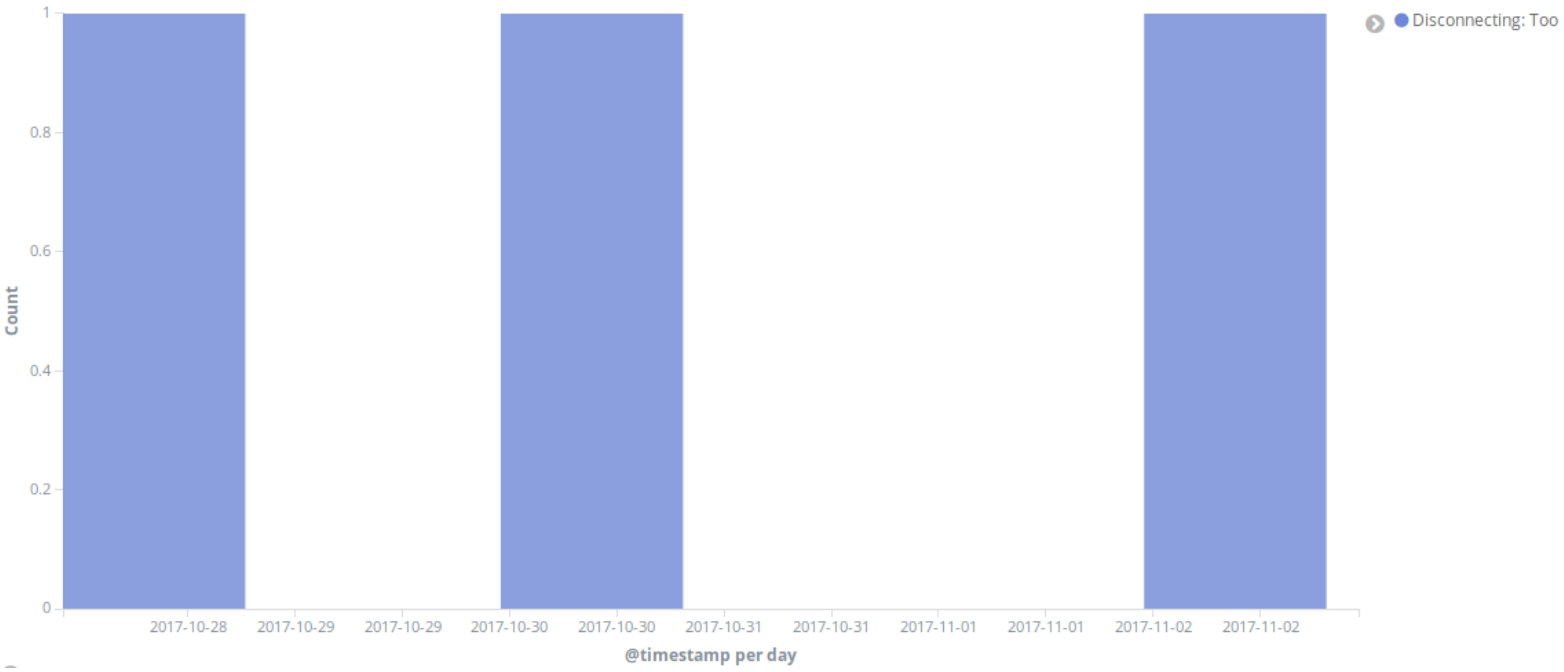

Figura 9- Número de autenticaciones Disconnecting Too Many por día

5.4.11. Geolocalización de las conexiones por países.

El siguiente gráfico muestra la geolocalización realizada a las conexiones recibidas a Guernika en el periodo desde la fecha 2017-02-01 a 2017-11-06. En el eje X se recoge el listado de países que al menos contienen una conexión y en el eje Y se recoge la suma total por país de estas conexiones.

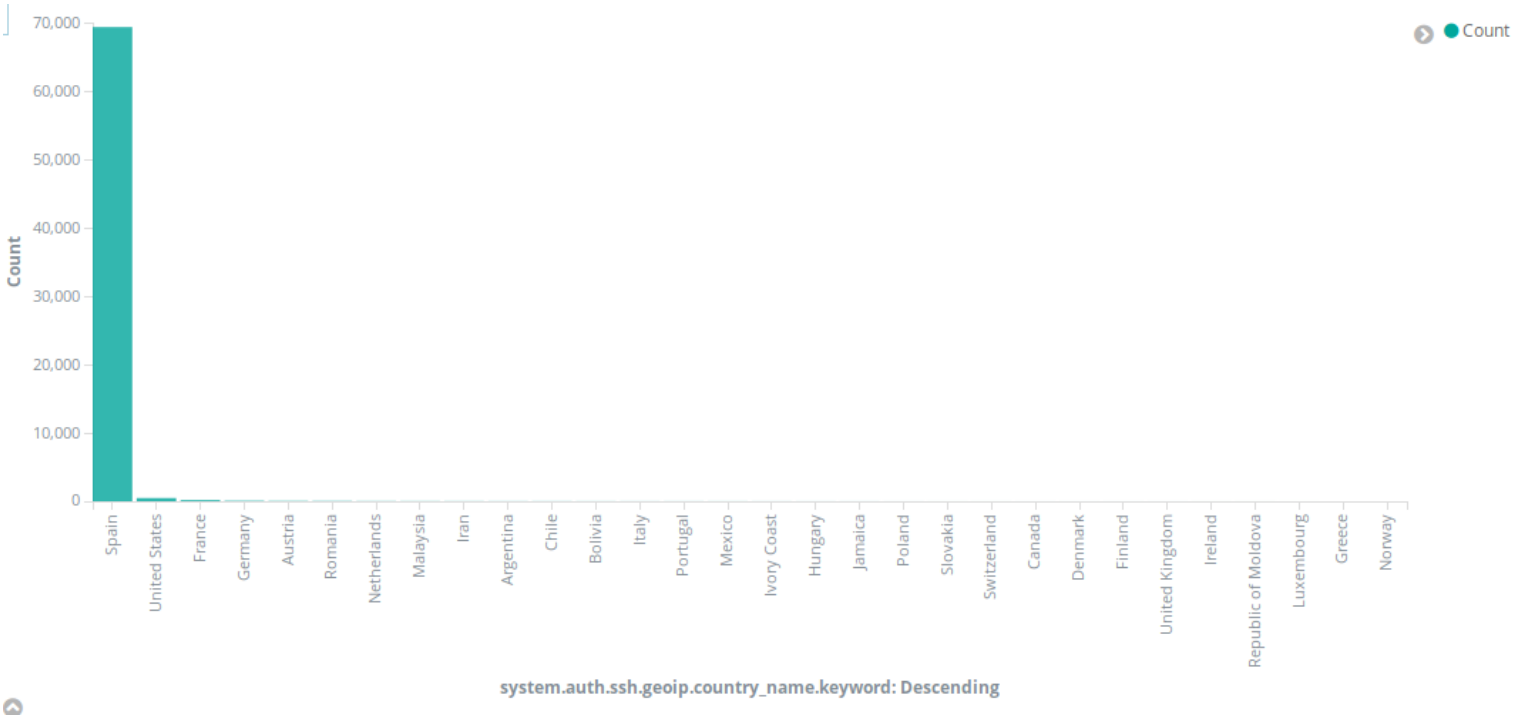

Figura 10- Número de conexiones por país

La tabla siguiente contiene el desglose del total de conexiones recibidas por Guernika por país.

PAIS	TOTAL CONEXIONES
Spain	69516
United States	475
France	178
Germany	82
Austria	54
Romania	47
Netherlands	31
Malaysia	27
Iran	23
Argentina	20
Chile	20
Bolivia	16
Italy	14
Portugal	14
Ivory Coast	11
Hungary	9
Jamaica	9
Poland	6
Slovakia	5
Switzerland	5
Canada	4
Denmark	4
Finland	4
United Kingdom	4
Ireland	3
Republic of Moldova	3
Luxembourg	2
Greece	1
Norway	1

Tabla 36- Número de conexiones por país

5.4.12. Geolocalización de las conexiones por continentes.

El siguiente gráfico de tarta muestra la geolocalización por continentes realizada a las conexiones recibidas a Guernika en el periodo desde la fecha 2017-02-01 a 2017-11-06.

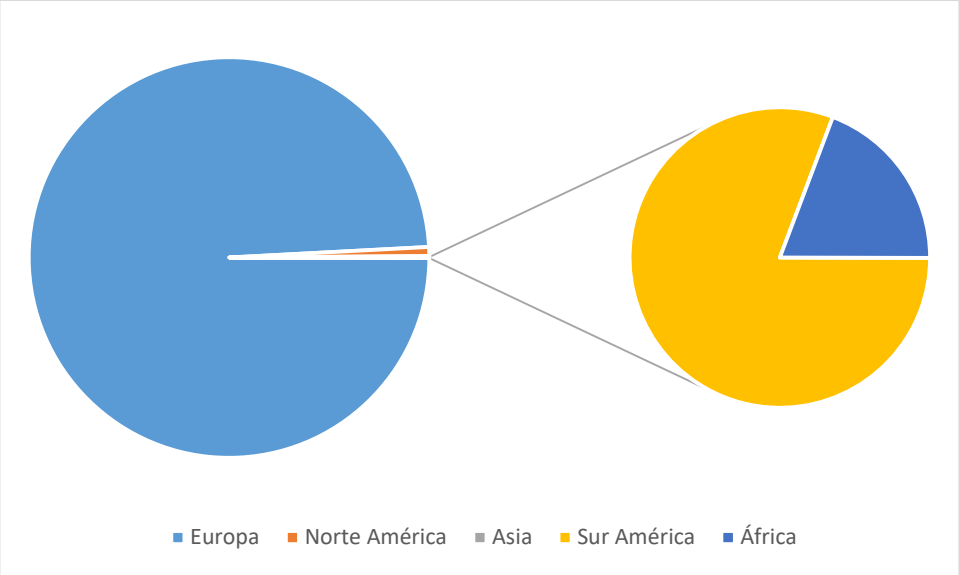

Figura 11- Número de conexiones por continente

La tabla siguiente contiene el desglose del total de conexiones recibidas por Guernika por país.

PAIS	TOTAL CONEXIONES
Europa	68,085
Norte América	498
Asia	50
Sur América	46
África	11

Tabla 37- Número de conexiones por continente

5.4.13. Geolocalización de las conexiones por ciudades.

El siguiente gráfico muestra la geolocalización realizada a las conexiones recibidas a Guernika en el periodo desde la fecha 2017-02-01 a 2017-11-06, debido al gran número de ciudades que compondrían el grafico solo se recogen las 25 con más visitas. En el anexo puede encontrarse el listado completo.

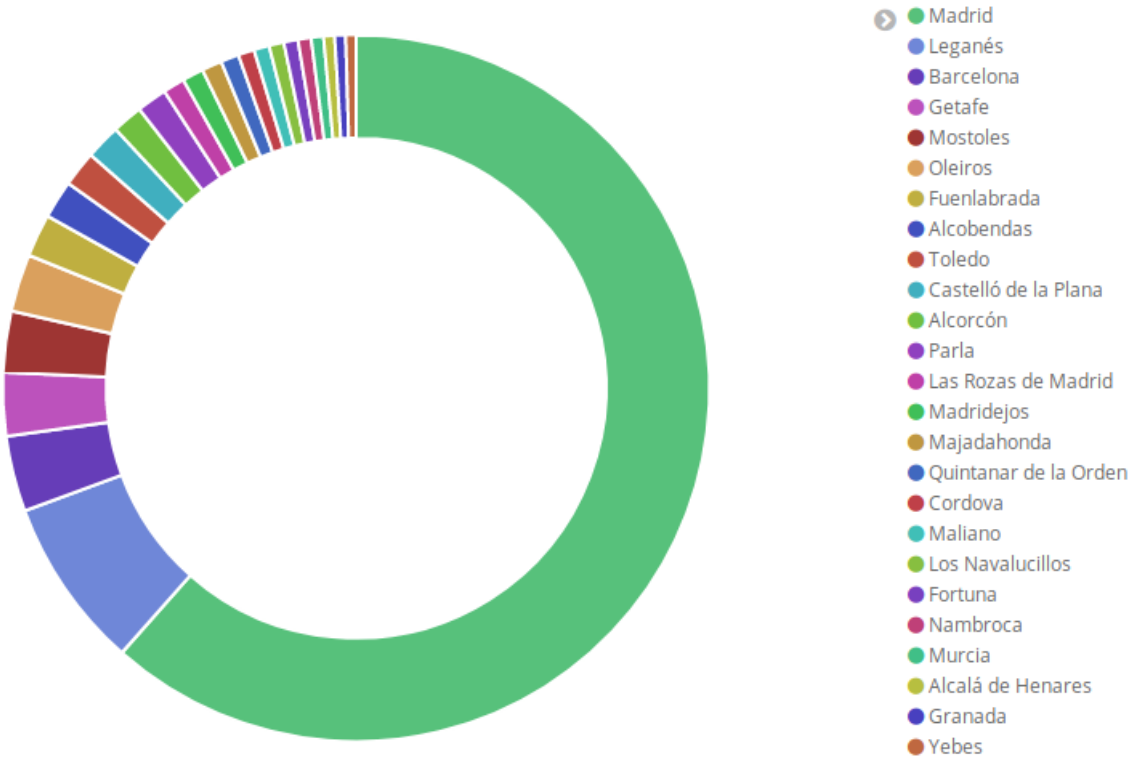

Figura 12- Número de conexiones por ciudades

La tabla siguiente contiene el desglose del total de conexiones recibidas por Guernika por ciudad.

Madrid	**36,966**
Leganés	4,698
Barcelona	2,083
Getafe	1,741
Mostoles	1,671
Oleiros	1,585
Fuenlabrada	1,177
Alcobendas	1,049
Toledo	980
Castelló de la Plana	967
Alcorcón	841
Parla	833
Las Rozas de Madrid	598
Madridejos	585
Majadahonda	553
Quintanar de la Orden	499
Cordova	444
Maliano	423
Los Navalucillos	408
Fortuna	389
Nambroca	361
Murcia	338
Alcalá de Henares	301
Granada	300
Yebes	284

Tabla 38- Top 25 ciudades con más conexiones hacía Guernika

5.4.14. Comandos utilizados con permisos de superusuario.

La siguiente tabla muestra los comandos utilizados con permisos de superusuario desde la fecha 2017-02-01 a 2017-11-06. La tabla se compone del comando utilizado junto con el número de veces que se ha efectuado tal comando.

Comando con ejecutado con permisos de superusuario	Total
/bin/su	12
/bin/bash	7
/bin/mkdir /usr/src/linux/dso	3
/usr/bin/apt-get install tree	3
/usr/bin/apt-get update	3
./ProgramaP	2
./compila.sh	2
/bin/chmod u+s /bin/ping	2
/bin/rm -R sumarMin	2
/bin/rm -fR /root .Trash	2
/usr/bin/apt-get install curl	2
/usr/bin/apt-get install glpk-utils	2
/usr/bin/gedit Makefile	2
./BUILD-FROM-SCRATCH --host=arm --build=arm	1
./a.out	1
./disk.dat	1
./evaluar_client locahost	1
./ex1seq.cpp	1
./filesystem.c	1
./filter.c -nofilter testfile.txt	1
./main	1
./rpc_server	1
/bin/chmod +x compilacion_client_UDP.sh	1
/bin/chmod 777 0356040302000e01	1
/bin/chmod 777 dssoo_p3_v4.2	1
/bin/ls	1
/bin/mkdir cosas	1
/bin/mount /mqueue	1
/bin/nc www.google.com	1
/bin/ping 163.117.142.237	1
/bin/rm -R 2	1
/bin/rm -r practica1	1
/bin/rm -rf unzip/	1

/bin/rm fake-page.txt	1
/bin/rmdir Práctica 1/	1
/bin/su cd 0555070804075405	1
/etc/init.d/core-daemon start	1
/usr/bin/apt-get install bsdgames	1
/usr/bin/apt-get install byacc flex	1
/usr/bin/apt-get install eclipse	1
/usr/bin/apt-get install git	1
/usr/bin/apt-get install glpk	1
/usr/bin/apt-get install go	1
/usr/bin/apt-get install openssl	1
/usr/bin/apt-get install rar	1
/usr/bin/apt-get install sl	1
/usr/bin/find	1
/usr/bin/find nbody	1
/usr/bin/gedit	1
/usr/bin/gedit .HelloWorld.c.sw	1
/usr/bin/gedit RR3.c	1
/usr/bin/gedit practica1.c	1
/usr/bin/gedit sumarMin.c	1
/usr/bin/git clone https://4a58075e0004@bitbucket.org/4a58075e0004/cluster-practice.git	1
/usr/bin/locate *.ino	1
/usr/bin/make	1
/usr/bin/make clean	1
/usr/bin/nano fake-page.txt	1
/usr/bin/nano ficheroA	1
/usr/bin/nano genetic-algorithm.go	1
/usr/bin/nano main	1
/usr/bin/passwd root	1
/usr/bin/pkill -KILL -u 50085600090c5c53	1
/usr/bin/scp /home/gonzalo/Descargas/filter.c 0309060a0a000201@guernika:~/	1
/usr/bin/scp 5207500300555253@guernika:~/Escritorio/lab1/p1.c /home/5b59160f5840	1
/usr/bin/scp weka.jar /users/alumnos-13-14/58565052060c5002/	1
/usr/bin/ssh f101	1
/usr/bin/vim compilacion_client_UDP.sh	1
/usr/bin/vim prueba	1

Escritorio	1
apk python	1
cd 5102535a030b5b01	1
cd 535159464705090c	1
0357535716	1
update	1
vii compilacion_client_UDP.sh	1

Tabla 39- Comandos utilizados con permisos de superusuario

5.4.15. Usuarios que no tienen permisos de superusuario tratando de ejecutar comandos como root.

La siguiente tabla muestra el listado de usuarios que han generado una incidencia en el sistema al tratar de ejecutar comandos con permisos de superusuario desde la fecha 2017-02-01 a 2017-11-06. La tabla se compone del usuario que ha generado la incidencia junto con el número de veces que se ha producido esta tipo de incidencia.

03020a5255010102	7
0402010204060554	6
0553025b03015704	4
5106065307530002	4
530002505e065600	3
0509510f54030402	3
0308020b5657510e	3
020150040d07560c	3
5005040d555a0552	2
5954070105540c04	2
58560b000b550105	2
550352010506040	2
5207500300555253	2
5356050356035353	2
0552060553055806	2
0309060a0a000201	2
055304560205000e	2
5652040356010107	1
0003070a090f0e53	1
5708035b5009570b	1
5153565450570d05	1
5105505157545653	1

5809520350075300	1
58070205530c0055	1
59560b070455540d	1
0551010301545200	1
00020408055b5700	1
50080557070f545b	1
03000b0800510057	1
520857025c525302	1
50085600090c5c53	1
5204500456000956	1
0206030b00020404	1
0356040302000e01	1
00040207020b0906	1
055600045404000d	1
5454030407505701	1
5652020b01535400	1
58000b06565d000	1
040301050b065704	1
902550254540702	1
0002500550065053	1
75503525d020605	1
0256515500570101	1
00540b53545b0705	1
5807065307050403	1
0705035d0556110a	1
575055074755500f	1
4a4a5b125a0c5849	1

Tabla 40- usuarios que han tratado de realizar sudo

5.4.16. Rutas desde la que se han tratado de ejecutar comandos como root.

La siguiente tabla muestra el listado de rutas del sistema desde las cuales se ha generado una incidencia en el sistema al tratar de ejecutar comandos con permisos de superusuario desde la fecha 2017-02-01 a 2017-11-06. La tabla se compone de la ruta desde donde se ha generado la incidencia junto con el número de veces que se ha producido esta tipo de incidencia.

/users/alumnos-10-11	**5**
/users/alumnos-15-16/0402010204060554	5
/users/alumnos-09-10/530002505e065600/Desktop/p1_planificador	3
/users/alumnos-12-13/0308020b5657510e/Desktop/EVALUABLE3	3
/users/alumnos-16-17/020150040d07560c	3
/	2
/users/alumnos-12-13/5151035809020f0f	2
/users/alumnos-12-13/0553025b03015704/Documentos/añopasado_dssoo_p3	2
/users/alumnos-12-13/0553025b03015704/Documentos/dssoo_p3_v4.2	2
/users/alumnos-12-13/58560b000b550105	2
/users/alumnos-12-13/5106065307530002	2
/users/alumnos-12-13/550352010506040/Documentos	2
/users/alumnos-15-16/0405030a04540608	2
/users/alumnos-15-16/0309060a0a000201	2
/users/alumnos-15-16/04050102560c5106	2
/users/alumnos-15-16/07060a5057505601/Documentos/oreja2	2
/users/alumnos-15-16/055304560205000e/practica1	2
/home	1
/media	1
/users	1
/users/adm/430340594f0e	1
/users/alumnos-09-10/0509510f54030402	1
/users/alumnos-09-10/0509510f54030402/Desktop/Forth/gforth-0.7.3	1
/users/alumnos-09-10/0509510f54030402/Desktop/dssoo_p3_v6	1
/users/alumnos-11-12/5005040d555a0552/Desktop/Pruebas	1
/users/alumnos-11-12/5005040d555a0552/Desktop/eq/Final	1
/users/alumnos-12-13	1
/users/alumnos-12-13/5652040356010107/Descargas	1
/users/alumnos-12-13/0003070a090f0e53/Desktop/detector-haart	1
/users/alumnos-12-13/5954070105540c04/Desktop/CP1/p1_planificador	1
/users/alumnos-12-13/5954070105540c04/Desktop/p1_planificador	1
/users/alumnos-12-13/5400035d04010250	1

/users/alumnos-12-13/5153565450570d05	1
/users/alumnos-12-13/5106065307530002/Descargas/compiling_debugging_dinami	1
/users/alumnos-12-13/5106065307530002/Descargas/documents-export-2016-02-26	1
/users/alumnos-12-13/5105505157545653	1
/users/alumnos-12-13/0055570854030e52/Documentos/aprendizaje	1
/users/alumnos-13-14/5809520350075300/dso-p1/P3/dssoo_fs	1
/users/alumnos-13-14/5207500300555253/Escritorio	1
/users/alumnos-13-14/5207500300555253/Escritorio/lab1	1
/users/alumnos-13-14/00020408055b5700/Descargas	1
/users/alumnos-13-14/50080557070f545b	1
/users/alumnos-13-14/50080557070f545b/Descargas	1
/users/alumnos-13-14/5356050356035353	1
/users/alumnos-13-14/520857025c525302	1
/users/alumnos-13-14/520857025c525302/go/src/mario/agalfa	1
/users/alumnos-14-15	1
/users/alumnos-14-15/5708035b5009570b/OS/Lab2/p2_minishell_2015	1
/users/alumnos-14-15/50085600090c5c53	1
/users/alumnos-14-15/5204500456000956/Documentos	1
/users/alumnos-14-15/5204500456000956/Escritorio	1
/users/alumnos-14-15/005252555507060d	1
/users/alumnos-14-15/0206030b00020404/Documentos	1
/users/alumnos-14-15/0356040302000e01/OS	1
/users/alumnos-14-15/055600045404000d/Documentos/FinalExamples	1
/users/alumnos-14-15/5454030407505701/pr-ia-2016-students	1
/users/alumnos-14-15/0552060553055806	1
/users/alumnos-14-15/03050a0709510057/Arquitectura	1
/users/alumnos-15-16/0007070807530302	1
/users/alumnos-15-16/58070205530c0055/p1_planificador_ultima	1
/users/alumnos-15-16/0551010301545200/dso	1
/users/alumnos-15-16/03000b0800510057	1
/users/alumnos-15-16/5806055302005e04	1
/users/alumnos-15-16/5806055302005e04/PROGRAMAS_EN_C	1
/users/alumnos-15-16/0051570508085507	1
/users/alumnos-15-16/00040207020b0906/SSOO/carpeta1	1
/users/alumnos-15-16/5652020b01535400	1
/users/alumnos-15-16/58000b06565d000/Practica1	1
/users/alumnos-15-16/040301050b065704/MPI_EXAMPLES	1
/users/alumnos-15-16/03020a5255010102	1
/users/alumnos-15-16/0002500550065053	1
/users/alumnos-15-16/5807065307050403/tema2	1

/users/alumnos-15-16/0402010204060554/evaluation	1
/users/alumnos-16-17	1
/users/alumnos-16-17/0256515500570101	1
/users/alumnos-16-17/00540b53545b0705	1
/users/alumnos-16-17/530201515c565b56	1
/users/profes	1
/users/profes/575055074755500f	1
/users/profes/4a4a5b125a0c5849/Documentos/ssoo/pr2_correccion/alumnos/g81	1
/users/windows/netlogon/POV-Ray/v3.7	1

Tabla 41- Rutas desde donde ha tratado de ejecutar comandos con sudo

5.4.17. Desglose de los comandos fallidos como root por usuario.

La siguiente tabla muestra el listado de comandos que se ha tratado de ejecutar en el sistema con permisos de superusuario desde la fecha 2017-02-01 a 2017-11-06 por cada usuario. En la tabla se recogerá los comandos ejecutados y el número de veces que se ha tratado de ejecutar.

Comandos	Usuario	Total
/usr/bin/gedit Makefile	530002505e065600	1
/usr/bin/make	530002505e065600	1
/usr/bin/make clean	530002505e065600	1
/bin/rm -fR /root .Trash	0509510f54030402	2
./BUILD-FROM-SCRATCH --host=arm --build=arm	0509510f54030402	1
./compila.sh	5652040356010107	1
/bin/chmod +x compilacion_client_UDP.sh	0308020b5657510e	1
/usr/bin/vim compilacion_client_UDP.sh	0308020b5657510e	1
vii compilacion_client_UDP.sh	0308020b5657510e	1
/usr/bin/git clone https://4a58075e0004@bitbucket.org/4a58075e0004/cluster-practice.git	0503005a0f080209	1
/usr/bin/find	5151035809020f0f	1
/usr/bin/find nbody	5151035809020f0f	1
./compila.sh	0003070a090f0e53	1
/bin/bash	0553025b03015704	1
./disk.dat	0553025b03015704	1
./filesystem.c	0553025b03015704	1
/bin/chmod 777 dssoo_p3_v4.2	0553025b03015704	1
./evaluar_client locahost	5005040d555a0552	1
./rpc_server	5005040d555a0552	1
/usr/bin/apt-get install byacc flex	5708035b5009570b	1
/usr/bin/gedit Makefile	5954070105540c04	1
/usr/bin/gedit RR3.c	5954070105540c04	1
/usr/bin/locate *.ino	5400035d04010250	1
/bin/chmod u+s /bin/ping	58560b000b550105	2
/bin/su	5153565450570d05	1
/bin/su	5106065307530002	2
./main	5106065307530002	1
/usr/bin/pkill -KILL -u 50085600090c5c53	5106065307530002	1

/bin/bash	5105505157545653	1
./ProgramaP	550352010506040	2
/usr/bin/apt-get update	0007070807530302	1
/usr/bin/apt-get install sl	5809520350075300	1
/usr/bin/nano main	58070205530c0055	1
/bin/bash	59560b070455540d	1
/usr/bin/apt-get install git	0551010301545200	1
/bin/bash	5207500300555253	1
/usr/bin/scp 5207500300555253@guernika:~/Escritorio/lab1/p1.c /home/5b59160f5840	5207500300555253	1
./ex1seq.cpp	00020408055b5700	1
/bin/su	50080557070f545b	1
/usr/bin/apt-get install tree	50080557070f545b	1
/usr/bin/gedit .HelloWorld.c.sw	5356050356035353	1
cd 535159464705090c	5356050356035353	1
/usr/bin/apt-get install tree	03000b0800510057	1
/bin/su	5806055302005e04	2
/usr/bin/apt-get install go	520857025c525302	1
/usr/bin/nano genetic-algorithm.go	520857025c525302	1
/bin/bash	020150040d07560c	1
/bin/rm fake-page.txt	020150040d07560c	1
/usr/bin/nano fake-page.txt	020150040d07560c	1
/usr/bin/gedit sumarMin.c	0051570508085507	1
/usr/bin/apt-get install openssl	50085600090c5c53	1
/etc/init.d/core-daemon start	5204500456000956	1
/usr/bin/apt-get install rar	5204500456000956	1
Escritorio	005252555507060d	1
/usr/bin/apt-get install tree	0356040302000e01	1
/bin/rm -R 2	00040207020b0906	1
/usr/bin/apt-get update	055600045404000d	1
apk python	5454030407505701	1
/usr/bin/apt-get update	0552060553055806	1
/bin/chmod 777 0356040302000e01	0552060553055806	1
/usr/bin/apt-get install glpk-utils	0405030a04540608	2
/bin/rmdir Práctica 1/	5652020b01535400	1
./filter.c -nofilter testfile.txt	0309060a0a000201	1
/usr/bin/scp /home/gonzalo/Descargas/filter.c 0309060a0a000201@guernika:~/	0309060a0a000201	1
/usr/bin/apt-get install glpk	04050102560c5106	1
update	04050102560c5106	1
/usr/bin/apt-get install bsdgames	58000b06565d000	1
/bin/ping 163.117.142.237	040301050b065704	1

/bin/su	03020a5255010102	2
/bin/su cd 0555070804075405	03020a5255010102	1
/usr/bin/passwd root	03020a5255010102	1
cd 5102535a030b5b01	03020a5255010102	1
0357535716	03020a5255010102	1
/bin/rm -R sumarMin	07060a5057505601	2
/usr/bin/vim prueba	0002500550065053	1
/usr/bin/gedit	055304560205000e	1
/usr/bin/gedit practica1.c	055304560205000e	1
/bin/mkdir cosas	75503525d020605	1
/bin/rm -r practica1	0256515500570101	1
/usr/bin/apt-get install eclipse	0256515500570101	1
/bin/nc www.google.com	00540b53545b0705	1
/usr/bin/apt-get install curl	5807065307050403	1
/bin/mkdir /usr/src/linux/dso	0402010204060554	3
/usr/bin/apt-get install curl	0402010204060554	1
/usr/bin/ssh f101	0402010204060554	1
/bin/mount /mqueue	0705035d0556110a	1
./a.out	575055074755500f	1
/bin/ls	095600	1
/bin/rm -rf unzip/	4a4a5b125a0c5849	1

Tabla 42- Total de comandos fallidos como sudo por usuario y cantidad de intentos

5.4.18. Usuarios que han ejecutado con éxito comandos con permisos de superusuario.

La siguiente tabla muestra el listado de usuarios que han ejecutado con éxito un comando en el sistema con permisos de superusuario desde la fecha 2017-02-01 a 2017-11-06. La tabla se compone del usuario que ha ejecutado el comando junto con el número de veces que lo ha conseguido.

snoopy	376768
CRON	245014
sshd	219274
systemd-logind	131484
systemd	16313
su	290
sudo	181
lightdm	56
gnome-keyring-daemon	44
login	38
dbus	30
rpcbind	17
passwd	12
polkitd(authority=local)	4
chsh	3
sftp-server	2

Tabla 43- usuarios que han ejecutado sudo con éxito

5.4.19. Did not receive identification string.

En el periodo que ocupa desde la fecha 2017-02-01 a 2017-11-06 se ha localizado el mensaje de error:

“0 *IPX*” su número de apariciones ha sido de un total de 324 apariciones.

couldn't communicate with already running daemon: Message did not receive a reply (timeout by message bus)

5.4.20. Inicios de sesión como superusuario.

La siguiente tabla muestra los inicios de sesión que se han producido en Guernika como superusuario. Se muestra el método de autenticación, la geolocalización de la IP, y el resultado de la autenticación.

Ciudad	País	Método de autenticación	Resultado de la autenticación	Total
Madrid	Spain	publickey	Accepted	139
Madrid	Spain	password	Failed	4
Dallas	United States	password	Failed	14
Ashburn	United States	password	Failed	13
Bucharest	Romania	password	Failed	9
Bucharest	Romania	Too many authentication failures	Disconnecting:	1
Budakeszi	Hungary	password	Failed	9
Viña del Mar	Chile	password	Failed	6
Arlington Heights	United States	password	Failed	5
Naples	Italy	password	Failed	4
Chisinau	Republic of Moldova	password	Failed	3
Frankfurt am Main	Germany	password	Failed	2
La Plata	Argentina	password	Failed	2
Los Angeles	United States	password	Failed	2
Mexico City	Mexico	password	Failed	2
Boardman	United States	password	Failed	1
San Antonio	United States	password	Failed	1

Tabla 44- Inicios de sesión como superusuario filtrado por método de autenticación resultado y total

5.4.21. Inicios de sesión como usuario admin.

La siguiente tabla muestra los inicios de sesión que se han producido en Guernika como el usuario admin. Se muestra el método de autenticación, la geolocalización de la IP, y el resultado de la autenticación.

Ciudad	País	Método de autenticación	Resultado de la autenticación	Total
Ashburn	United States	password	Failed	18
Madrid	Spain	password	Failed	6
Arlington Heights	United States	password	Failed	3
San Antonio	United States	password	Failed	3
Amsterdam	Netherlands	password	Failed	3
Hoest	Germany	password	Failed	2
La Plata	Argentina	password	Failed	2
Washington	United States	password	Failed	2
Boardman	United States	password	Failed	1
Bucharest	Romania	password	Failed	1
Snina	Slovakia	password	Failed	1

Tabla 45- Intentos de inicio de sesión como usuario admin

5.4.22. Inicios de sesión como usuario guest.

La siguiente tabla muestra los inicios de sesión que se han producido en Guernika como el usuario guest. Se muestra el método de autenticación, la geolocalización de la IP, y el resultado de la autenticación.

Ciudad	País	Método de autenticación	Resultado de la autenticación	Total
Ashburn	United States	password	Failed	2
Arlington Heights	United States	password	Failed	1
San Antonio	United States	password	Failed	1

Tabla 46- Inicios de sesión como usuario Guest

Capítulo 6: Informe final

Este capítulo trata sobre la evaluación de la información que se encuentra en los logs. En él se extraen las conclusiones acerca de la información recogida y además recoge un informe de seguridad sobre el estado de Guernika en base al estudio realizado.

6.1. Fuente de información.

La fuente de información con la que se ha elaborado el estudio ha sido la colección de logs recibida. En concreto utiliza los ficheros auth.log.

6.2. Objetivo del informe.

- Evaluar el impacto de la exposición de Guernika a intenet a través de su servicio SSH.
- Realizar un análisis sobre el comportamiento de los usuarios en Guernika.

6.3. Periodo de tiempo y contexto en el que se enmarca el estudio.

El periodo de tiempo como se anticipaba en el capítulo anterior abarca desde la fecha 2017-02-01 hasta 2017-11-06.

Es muy importante tener en mente el periodo de tiempo evaluado, porque este servidor en concreto se trata de un servidor muy característico, ya que está ubicado en la universidad Carlos III de Madrid y es utilizado con frecuencia durante el curso.

Precisamente esta afirmación es clave para entender lo que se observa a través de los logs durante el estudio, pues se trata de un estudio que abarca 2 cuatrimestres, es decir un curso de universidad.

Por lo que el estudio permite obtener que ha ocurrido durante un curso con sus dos cuatrimestres.

6.4. Contraste de hipótesis

En este punto se recoge la valoración de lo que se esperaba encontrar en Guernika frente a las evidencias encontradas.

Usuarios: Como se especulaba en el punto *5.2 Hipótesis sobre los tipos de usuario y comportamiento esperados* se han identificado distintos tipos de usuarios. En este caso la hipótesis ha sido cierta pues se han encontrado usuarios identificados a través de un NIA, el usuario administrativo root y usuarios específicos para el funcionamiento de programas.

No obstante sí se han encontrado usuarios no identificados por NIA si no por un nombre de usuario.

Comportamientos: Tal y como refleja el apartado *5.2.2 Hipótesis sobre la clasificación de comportamientos esperados* se esperaba usuarios que tuvieran comportamientos normales, es decir que no realicen actividades sospechosas o fuera de lo que se esperaría de una cuenta destinada a un alumno. Como se anticipaba al estudio si se han localizado comportamientos sospechosos pero no se ha encontrado una evidencia potente para catalogar a los usuarios existentes de atacantes. Por otro lado si se han encontrado evidencias sobre ataques a Guernika por parte de usuarios no registrados en el equipo.

Ataques: Como refleja el punto *5.2.3 Hipótesis sobre los ataques que se espera encontrar* se han encontrado ataques hacía Guernika por lo que la hipótesis ha sido cierta. No obstante no se ha encontrado la evidencia de un ataque potente hacía el servidor.

Fases de un ataque: Como refleja el punto *5.2.5 Hipótesis sobre posibles evidencias de ataques* se puede ubicar los ataques recibidos como *Reconomiento*. Por lo que la hipótesis se cumple.

6.5.Programas que escriben en el fichero auth.log

Refiriéndose al punto *5.4.2 Programas que han escrito en el log* se obtiene la lista de programas que han escrito en todos los logs auth.log. Esto es importante porque se obtiene que tipo de registros y que tipo de información se recoge en los logs,

Se observa que el programa que más ha escrito en los logs se trata de ***Snoopy***. Snoopy se trata de una herramienta de monitorización de comandos. Obtiene aquellos comandos que se han ejecutado en el sistema, funciona de forma similar a *bashhistory* pero mediante la captura de exec() y execv() [29], esto hace que pensar de que se trate del programa que más ha escrito en el fichero tenga sentido, ya que son muchos usuarios los que inician sesión durante el cuatrimestre y ejecutan por tanto comandos de forma constante.

Mediante la observación del repositorio de código alojado en github se confirma que efectivamente se escribe en el fichero *auth.log.*

Si bien el programa permite capturar los comandos ejecutados, no se trata de un método fiable frente a usuarios maliciosos debido a que el mismo autor incide en que su herramienta puede ser alterada para no registrar comandos, la información está recogida en la FAQ del proyecto [30].

Aclara además el método de saltarse la restricción de que un usuario deje de estar trazado a través de Snoopy. [30]

Si bien el programa resulta útil para capturar el comportamiento de la mayoría de los usuarios, se debe de tener en cuenta que un usuario avanzado puede haber evitado sido trazado por el programa.

Por otro lado observando el resto de programas se descubre que gran parte de los programas que escriben en el fichero, son partes del sistema operativo, que se trata de algo esperable. Programas como ***cron*** que se trata de un demonio para ejecutar tareas programadas, ***systemd*** y ***system*** demonios de administración y servicios del sistema operativo, ***lightdm*** que se trata del gestor de inicio de sesión gráfico que en general se correspondería con los accesos que se realizan mediante la interfaz gráfica así como ***dbus*** que se trata de un sistema de comunicación y sincronización entre procesos.

Se encuentran también los gestores de superusuario ***su y sudo***, su análisis será importante pues *su* se utiliza para iniciar sesión como *root* y *sudo* para que el usuario que ejecuta un comando obtenga los permisos equivalentes a utilizar el usuario administrativo *root.* Que aparezca en el log este tipo de entradas se considera normal pues recoge la traza realizada por este tipo de comandos que forman parte de la administración del equipo, a lo que si se debe de prestar atención es al uso de los comandos. Cuyo estudio se recogerá más adelante.

Observando finalmente los programas que menos han escrito en el log se observan los siguientes:

passwd, polkitd y sftp-server. Estos llaman la atención debido a la poca frecuencia con la que aparecen escribiendo en el log. Se estudiarán más delante de forma individual a través de los usuarios que los ejecutaron.

6.6.Métodos de autenticación.

Como se observa en el apartado *5.4.3 Método de autenticación* los métodos para iniciar sesión en Guernika prácticamente en su totalidad, 98% sobre el total, están compuestos mediante la identificación de usuario y contraseña. Esto es el comportamiento esperado por parte de los usuarios como medio para acceder a Guernika fuera de las aulas como se recoge en el manual de la universidad [31].

Llama la atención por tanto que exista un 1,59% que accede mediante clave pública. Un estudio más exhaustivo de las conexiones que se producen mediante este método de autenticación arroja que la mayoría de los inicios de sesión producidos son realizados mediante clave pública por el usuario *root*, el resto se trata de conexiones realizadas por alumnos.

Esto es indicador de que el usuario *root* es compartido por distintos administradores y se utiliza como método de autenticación alternativo a la contraseña. Ofreciendo un método de control de inicio de sesión más robusto que la contraseña. No obstante también indica que el usuario root se encuentra habilitado mediante ssh.

Aunque se encuentra más protegida la cuenta de *root* al utilizar la clave pública, se recomienda no permitir realizar inicios de sesión como *root* directamente a través de *ssh* si se puede evitar, pues se desconoce el funcionamiento y operativa para la que se realiza este inicio de sesión como root. En su lugar se recomienda utilizar un usuario intermedio que inicie sesión a través de ssh y después este habilitado poder ejecutar el comando *su* que le permita iniciar sesión como *root*. Esto impide que cualquier ataque directo contra el inicio de sesión como root no pueda realizarse y en su lugar también tenga que conocerse las credenciales del usuario intermedio.

Respecto a los alumnos que utilizan cómo método de autenticación la clave pública, probablemente se trata de configuraciones personalizadas y se encuentren cada clave configurada en su directorio *home.* Se recomienda considerar si se desea permitir este método de autenticación a todos los usuarios.

Se observa que aparece *too many authentication failures* que podría indicar ataques de fuerza bruta pero se descarta que sea un ataque de fuerza bruta debido a las pocas apariciones de este mensaje.

6.7.Estado de la autenticación.

Atendiendo al apartado *5.4.4 Estado de la autenticación* se observa que el 93% de los intentos de inicio de sesión se han producido con éxito. Esto se trata de algo normal pues se espera un número alto de inicios de sesión de forma correcta.

Por otro lado se observa que un 5,29% se han registrado como *failed* y el 1,38% se han registrado como *invalid*.

La diferencia entre *failed* e *invalid* es que failed se refiere a un intento de inicio de sesión que no se ha llegado a producir por una identificación incorrecta mientras que invalid se debe por intento de inicio de sesión de un usuario inválido para iniciar sesión, como podrían ser usuarios creados para realizar tareas de aplicaciones en el equipo como por ejemplo el usuario que se crearía para manejar exclusivamente MySQL.

Finalmente respecto a los estados *Disconnecting y Disconnecting: Too many authentication failures for invalid* problamente se trata ataques de fuerza bruta en el que se prueba un inicio de sesión observando el resultado o bien se trata de demasiados intentos de inicio sesión en un número de inicios de sesión inusualmente alto.

El contexto en el que se ubica el servidor Guernika se ve claramente reflejado en el comportamiento que se observa en el tiempo respecto del resultado del intento de inicio de sesión en el equipo. Este comportamiento puede observarse con claridad en el punto *5.4.5 Estado de la autenticación por días*, que incluye todos los estados recogidos de la autenticación por días.

Observando este punto se observa claramente el desarrollo del cuatrimestre mediante el acceso de los usuarios a Guernika por SSH.

Se puede deducir que los picos de conexiones que se producen durante el año coinciden con la realización de prácticas en la universidad. Concentrando el mayor número de peticiones en las fechas coincidentes de finales de cuatrimestre, siendo el primer cuatrimestre desde septiembre a diciembre donde se observa un número alto de conexiones a medida que se acerca en el tiempo Diciembre y en el segundo cuatrimestre desde Enero a Mayo, encontrando que coincide durante Mayo el mayor número de peticiones en el segundo cuatrimestre.

La visualización de forma independiente de los resultados de la autenticación por días de todos los estados recogidos permite obtener las siguientes conclusiones:

Atentiendo de forma exclusiva a las peticiones aceptadas *5.4.8 Estado de la autenticación filtrado por el estado Accepted por días*

Se observan tres picos predominantes de actividad coincidiendo con las fechas descritas anteriormente de máxima actividad del cuatrimestre. Esto tiene sentido debido a que el periodo de mayor uso de la plataforma.

De igual modo para los resultados de inicio de sesión con el estado failed se observa la misma tendencia *5.4.6 Estado de la autenticación filtrado por el estado Failed en días* pues se achaca a errores humanos al introducir las credenciales.

No obstante si se observa con más detenimiento la fecha que abarcaría el mes de abril se observa cómo se sale de un número muy alto de inicios de sesión y despúes la tendencia se encuentra cercana a 0, tanto como para los inicios de sesión de forma exitosa como los que se han reportado el estado *Failed.*

Esto puede indicar o bien un fallo en la recolección de los Logs o bien un fallo o interrupción del servicio durante el mes de abril. En caso de que no se tuviera consciencia de este comportamiento se recomienda revisar en detalle lo ocurrido durante este periodo de tiempo.

Por otro lado si se observa el punto *5.4.9 Estado de la autenticación filtrado por el estado Disconnecting por días* y *5.4.10 Estado de la autenticación filtrado por el estado Disconnecting: Too many authentication failures for invalid* se observa como los resultados de esta autenticación se suceden en los periodos de mayor actividad con la excepción de Julio y Agosto donde se conoce que los usuarios típicos de Guernika, los estudiantes, se encuentran de vacaciones al ser periodo estival. En este periodo se observa además como la suma de los estados distintos a *Accepted* se han sucedido con más frecuencia.

Si se compara además este periodo con los intentos de inicio de sesión con el estado Failed se descubre que además se han obtenido los mismos resultados que un día de máxima intensidad con multiples usuarios en la universidad.

Lo cual indica que se ha sucedido un ataque de fuerza bruta con el fin iniciar sesión de forma no autorizada.

Un análisis del punto *5.4.7 Estado de la autenticación filtrado por el estado Invalid en días* permite obtener conclusiones semejantes, durante el periodo del cuatrimestre en las fechas cercanas al final de cuatrimestre se concentran un gran número de peticiones pero en el periodo estival se han sucedido también un número importante de peticiones con picos cercanos a periodos de alta actividad.

Esto refuerza la idea del ataque de fuerza bruta.

6.8. Origen de las conexiones.

Utilizando el estudio de geolocalización realizado con los puntos: *5.4.11 Geolocalización de las conexiones por países., 5.4.12 Geolocalización de las conexiones por continentes.5.4.13 , Geolocalización de las conexiones por ciudades.*

Se observa que el mayor número de peticiones son correspondientes a España. Lo cual es normal pues se trata de una universidad ubicada en territorio español.

El desglose de las peticiones por ciudades también revela que la ciudad que aglutina el mayor número de peticiones es Madrid, también tiene sentido al ser la comunidad autónoma en la que está ubicada la universidad Carlos III.

No obstante se observa una carencia de peticiones de países asiáticos, sobre todo aquellos desde los que se espera una gran cantidad de peticiones como podría ser Rusia y China al ser conocidos por ser los países que más ciberataques realizan a España [32]. Durante el periodo evaluado no se observa ninguna petición con estos orígenes y si se observa más detenidamente el listado se observa que las peticiones no proceden de ningún otro país asiático con la excepción de Irán y Malasia.

Esto lleva a pensar que Guernika se encuentra detrás de un cortafuegos que limita las peticiones por su origen de conexión en el globo. Probablemente el cortafuegos no esté ubicado en Guernika si no que estará detrás de otro tipo de infraestructura pues no se ha encontrado de ninguna evidencia de direcciones IP descartadas por Guernika.

6.9. Análisis del uso y acceso a permisos administrativos

Analizando los puntos:

- *5.4.14 Comandos utilizados con permisos de superusuario.*
- *5.4.15 Usuarios que no tienen permisos de superusuario tratando de ejecutar comandos como root.*
- *5.4.16 Rutas desde la que se han tratado de ejecutar comandos como root.*
- *5.4.17 Desglose de los comandos fallidos como root por usuario.*
- *5.4.18 Usuarios que han ejecutado con éxito comandos con permisos de superusuario.*

Se observa que los comandos más utilizados han sido para obtener una Shell con permisos de root. Esto puede tener el origen en intentar resolver un problema para el que no se tenía suficientes privilegios, un usuario administrador del sistema realizando tareas o bien se trata de una escalada de privilegios maliciosa.

Prestando atención a los comandos menos utilizados se puede obtener la siguiente información:

Se ha intentado instalar paquetes nuevos en el equipo o actualizar el sistema operativo sin tener las credenciales para ello, esto no se identifica con un comportamiento anómalo si no tal vez con problemas al ejecutar determinados programas, probablemente por falta de dependencias para intentar hacer funcionar algún programa.

Por otro lado si se han encontrado algunos comportamientos anómalos como:

/usr/bin/ssh f101 intento de iniciar sesión ssh en el equipo desconocido para el análisis *f101*.

/usr/bin/find nbody desde el directorio raíz, intento de buscar la práctica nbody.

/bin/chmod 777 dssoo_p3_v4.2 intento de modificar los accesos a todos los permisos para cualquiera a la carpeta que se presupone contiene una práctica.

/bin/chmod u+s /bin/ping intento de modificación de los permisos del binario ping.

/usr/bin/pkill -KILL -u 50085600090c5c53 finalización de programas de usuarios de usuarios.

/etc/init.d/core-daemon start intento de inicio del demonio CORE.

/usr/bin/passwd root intento de cambio de contraseña al usuario administrativo root.

Aunque se identifican como comportamientos maliciosos o alertas, no se trata de alertas relacionadas con un ataque premeditado coordinado si no que se identifican como alumnos probando que pueden realizar o no, además de tratar de obtener acceso a carpetas de algún otro alumno.

El número total de usuarios que han generado una alerta en el equipo por tratar de utilizar root o sudo es de 49 usuarios únicos. Se trata de un número muy pequeño en proporción con el total de usuarios en el equipo por lo que se deduce que el comportamiento malicioso a través de root no es habitual.

Por otro lado identificando las rutas desde la que se han lanzado algunos comandos se ha obtenido que algunos usuarios tienen acceso a determinadas carpetas de alumnos ya que se encuentran en el path desde donde lanzaron el comando, lo cual indica que algunas carpetas de usuario se encuentran abiertas a otros usuarios.

Esto no tiene por qué ser una incidencia generada por el sistema si no que puede tratarse de una mala configuración de permisos por parte de un usuario o tal vez un error heredado.

Se recomienda revisar los permisos de las carpetas que se encuentran en el directorio */home.*

Respecto a los usuarios que han logrado ejecutar con éxito comandos como *root* se dispone del listado del punto *5.4.18 Usuarios que han ejecutado con éxito comandos con permisos de superusuario.*

Se observa que los usuarios que han conseguido ejecutar comandos con permisos de *superusuario* han sido programas del equipo y el propio usuario administrador del equipo, lo cual es correcto y una buena señal siempre y cuando no se haya producido ningún acceso no autorizado.

6.10. Fuerza bruta y spam de bots

Teniendo en cuenta la información que se encuentra en los puntos *5.4.19 Did not receive identification string.*,*5.4.20 Inicios de sesión como superusuario.*, *5.4.21 Inicios de sesión como usuario admin.*, *5.4.22 Inicios de sesión como usuario guest.*

Se observa que efectivamente llegan ataques de fuerza bruta a Guernika por medio de conexiones que han pasado el filtro de geolocalización sobre el que está ubicado Guernika. Se observa que la principal fuente de ataques es Estados Unidos, se trata de una zona geo-política difícil de filtrar pues Estados Unidos goza de reconocimiento internacional, aunque es una fuente importante de ataques importante en la red. Los intentos de inicio de sesión están relacionados con el mensaje *couldn't communicate with already running daemon: Message did not receive a reply (timeout by message bus)* pues tratan de obtener el estado que ha producido un intento de inicio y cortan la comunicación antes de tiempo, se ha observado en los logs que este tipo alertas preceden a intentos de inicio con un usuario en concreto que falla en su autenticación.

No obstante se considera positivo el funcionamiento del filtro que se realiza antes de que las conexiones lleguen a Guernika.

Pues no existe ningún sistema perfecto que evite por completo todas las amenazas y por tanto es normal que lleguen algunas de ellas a efectuarse, pero por otro lado muchas de las direcciones IP desde las que ha sufrido ataques de fuerza bruta Guernika para tratar de averiguar la contraseña de root, admin o guest se encuentran en listas negras accesibles desde internet como *mxtoolbox.com* [33] se recomienda tener en cuenta estos repositorios de listas negras de direcciones IP si no se hace ya y revisar que el usuario admin y guest no se encuentren activados en el equipo.

Este tipo de ataques se considera que han sido autónomos es decir realizados por bots que escanean webs de forma automatizada y no se trata de ataques coordinados de forma efectiva.

6.11. Resultado final sobre el informe

El informe realizado sobre los ficheros *auth.log* permite concluir que el estado de las conexiones SSH estudiadas **no** reportan la evidencia de un ataque grave frente a Guernika.

No obstante se recomienda tener las siguientes consideraciones:

1. Considerar auditar los comandos y actividad de los usuarios complementando la configuración existente de *Snoopy* con la herramienta de *Red Hat* disponible para las distribuciones *Linux* ***auditd*** [34]
2. Considerar deshabilitar el usuario *root* mediante acceso directo a *SSH*, se propone habilitar el acceso *root* a través de otro usuario intermedio.
3. Revisar los accesos mediante la clave pública tanto de *root* como del resto de usuarios.
4. Revisar la actividad del usuario *root*.
5. Tener en consideración y revisar el periodo de inactividad del servidor durante el mes de abril.
6. Revisar los permisos de las carpetas *home* los usuarios.
7. Revisar los permisos de algunos binarios importantes, que no se encuentren accesibles con permisos públicos.
8. Tener en cuenta los accesos remotos por bots y considerar trazar las IPS recibidas con listas negras.
9. Considerar si no se utiliza, el uso de *rkhunter* con el fin de buscar rootkits y posibles amenazas y contrastar con los comandos trazados durante el proyecto.

Capítulo 7: Gestión del proyecto

En este punto se recoge la planificación realizada en las distintas fases involucradas así como el coste imputable total desglosado.

7.1.Planificación

Aquí se recoge la planificación que se ha realizado durante el proyecto realizado.

7.1.1. Duración total del proyecto y esfuerzo

2 meses, en días naturales. Se estima que el esfuerzo dedicado por día será de ente 5 y 6 horas al día.

7.1.2. Fases del proyecto

1) Análisis
2) Preparación del entorno
3) Test y validación de las herramientas
4) Carga de datos
5) Conclusiones
6) Documentación

7.1.3. Tiempo por fases

La siguiente tabla recoge el tiempo en días por cada fase.

Fase	Duración en días
Análisis	16 días
Preparación del entorno	4 días
Test y validación de las herramientas	1 día
Carga de datos	4 días
Conclusiones	12 días
Documentación	17 días
Total	**54 días**

Tabla 47- Inicios de sesión como usuario Guest

7.1.4. Fase 1: Análisis

Duración 16 días.

Se trata de la fase que involucra tanto el análisis del problema como el análisis y estudio de las herramientas que se utilizan en el proyecto.

- Análisis del problema.
- Análisis de las posibles soluciones.
- Búsqueda de alternativas.
- Estudio de las herramientas involucradas.
- Diseño de la arquitectura para resolver el problema.
- Diseño de arquitecturas alternativas.

7.1.5. Fase 2: Preparación del entorno

Duración 4 días.

Se trata de la fase en la que se aplica el diseño realizado y se preparan las herramientas para su utilización.

- Instalación del sistema operativo.
- Instalación de las dependencias software necesarias para el manejador de logs.
- Instalación del manejador de Logs.
- Ajuste y configuración del manejador de Logs.

7.1.6. Fase 3: Test y validación de las herramientas

Duración 1 día.

Se trata de la fase en la que se valida y comprueba que el entorno instalado es el que se corresponde con el diseño arquitectónico realizado.

7.1.7. Fase 4: Carga de datos

Duración 4 días.

Se trata de la fase en la que se cargan los Logs a la plataforma y se valida que la calidad de los datos que se encuentran cargados en la plataforma es correcta.

- Carga de los Logs.
- Verificación de la carga de datos.

7.1.8. Fase 5: Conclusiones

Duración 12 días.

Se trata de la fase en la que se estudian y analizan los datos obtenidos.

- Identificación de usuarios.
- Geolocalización de usuarios.
- Seguimiento del comportamiento durante la sesión.
- Trazabilidad de los comandos utilizados.
- Comprobación de intentos de ataque.
- Clasificación de usuarios.
- Descarte de falsos positivos.

7.1.9. Fase 6: Documentación y ofuscación de los datos

Duración 17 días.

Se trata de la fase en la que se deja constancia de forma detallada del estudio realizado.

- Redacción de la memoria del proyecto.
- Conclusiones finales.

7.1.10. Planificación prevista

ACTIVIDAD	PLAN START	PLAN DURACION	ACTUAL START	ACTUAL DURACION	PERCENT COMPLETADO
Análisis	1	16	1	16	100%
Preparación del entorno	17	4	17	4	100%
Test y validación	21	1	21	1	100%
Carga de datos	22	4	22	4	100%
Conclusiones	26	12	26	12	100%
Documentación	38	17	38	17	100%

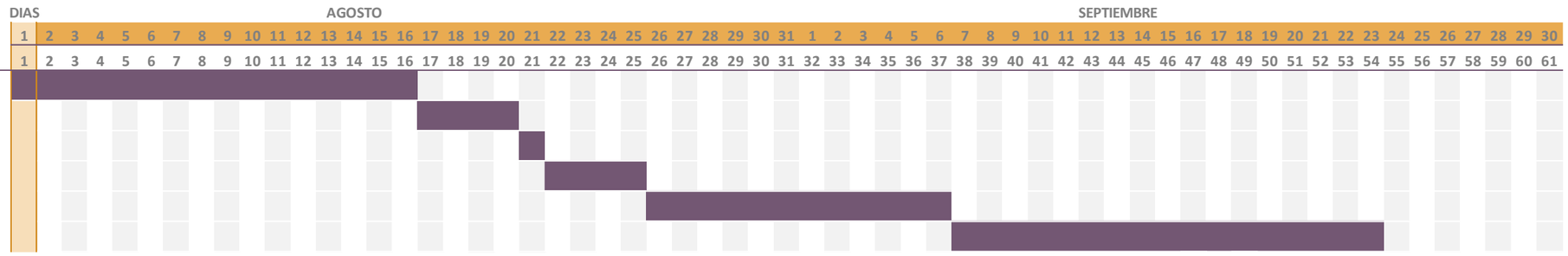

Ilustración 25- Planficación prevista durante el proyecto

7.1.11. Comparación entre la planificación prevista y el tiempo empleado

ACTIVIDAD	PLAN START	PLAN DURACION	ACTUAL START	ACTUAL DURACION	PERCENT COMPLETADO
Análisis	1	16	1	19	100%
Preparación del entorno	17	4	17	3	100%
Test y validación	21	1	21	2	100%
Carga de datos	22	4	22	6	100%
Conclusiones	26	12	26	9	100%
Documentación	38	17	38	18	100%

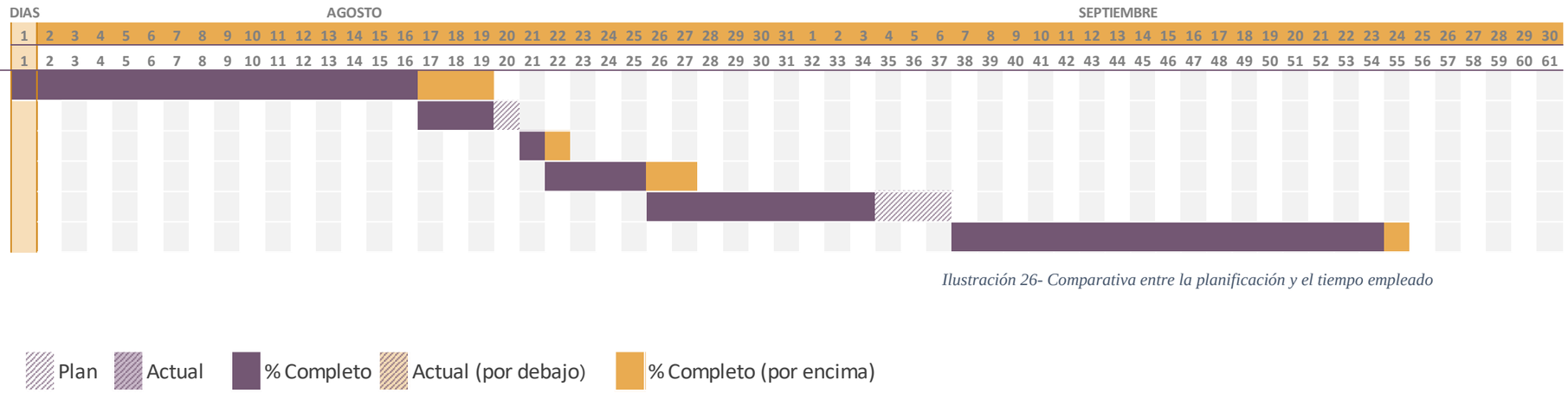

Ilustración 26- Comparativa entre la planificación y el tiempo empleado

7.2.Presupuesto

En este punto se recoge de forma detallada el presupuesto necesario para la realización del proyecto.

7.2.1. Clasificación del personal

Para la realización del proyecto se precisa de la colaboración en equipo de varias personas, la tutora de proyecto y el desarrollador/analista del proyecto.

Representando estos dos puestos distintos en el proyecto, se ha utilizado la siguiente tabla:

Categoría 1	Categoría 2
Tutora de proyecto	Desarrollador/analista

Tabla 48- Categorías profesionales en el proyecto

7.2.2. Presupuesto para el personal

	Categoría 1	Categoría 2
Salario bruto al año	38000 €	25000 €
Seguridad social al año	12540 €	8250 €
Coste total	50540 €	33250 €
Coste por hora	29,64€	19,50€
Horas trabajadas	16	306
Coste Imputable	474,24 €	5967 €
		TOTAL: 6441,24€

Tabla 49-Coste del personal

Formula del coste por hora:

*Coste total / (155horas/mes * 11meses)*

7.2.1. Presupuesto para el equipo

Descripción	Coste	Uso	Vida útil	Depreciación anual	Depreciación al mes	Coste imputable
Monitor 22'	125€	2 meses	36 meses	41,66€	3,47€	6,94€
Ordenador	982€	2 meses	36 meses	327,3€	27,27€	54,54€
Impresora Brother HL-L5200DW	236,99€	1 semana	36 meses	78,99€	6,58€	13,16€
					TOTAL	**77,64€**

Tabla 50-Coste del equipo

Para el cálculo de la depreciación se ha utilizado el método de depreciación en línea recta.

Depreciación anual: Valor del activo/3 años

Depreciación mensual: Valor del activo/36 meses

7.2.2. Presupuesto para fungibles y otros gastos

Descripción	Coste imputable
Folios	6,54€
Bolígrafos	8€
Recambios impresora	34,99€
Agua, luz e internet	99€ mes, 198€ dos meses
Alquiler local	450€ mes, 900€ dos meses
Licencia software: Debian 9	0€
Licencia software: Elastic Stack	0€
TOTAL	**1147,53€**

Tabla 51-Presupuesto para fungibles y otros gastos

7.2.3. Coste imputable total

Coste total imputable destinado al personal: **6441,24€**

Coste total imputable destinado al equipo: **77,64€**

Coste total imputable destinado para fungibles y otros gastos: **1147,53€**

Coste total: **7666,41€**

Descripción	Coste imputable
Coste del personal	6441,24€
Coste del equipo	77,64€
Fungibles y otros gastos	1147,53€
Costes indirectos 21%	1609,94€
Beneficio 25%	2319,08€
IVA 21%	2435,04€
TOTAL	**14029,97€**

Tabla 52-Coste imputable total

Por lo que el coste total será de **catorce mil veintinueve euros con noventa y siete céntimos.**

7.2.4. Impacto socio-económico

Aparte del coste y beneficio económico descrito anteriormente que se obtendría al realizar este proyecto, también tendría una repercusión social y ética directa sobre la universidad pública y por tanto de todos los ciudadanos:

1. Incrementaría la detección y prevención de amenazas informáticas, lo que contribuye a mantener el prestigio y confianza en la universidad.
2. Vela por la privacidad de los alumnos al contribuir a hacer parte de la universidad más robusta ante posibles fugas de información sobre los alumnos.
3. Técnicamente se trata de una mejora de la capacidad de análisis de la seguridad del servidor.
4. Contribuye a detectar actividades fraudulentas.

Capítulo 8: Conclusiones y líneas futuras

En este capítulo se recogen las posibles mejoras como continuación del proyecto y los problemas principales encontrados en la realización del proyecto.

8.1. Conclusiones

Se consideran las siguientes conclusiones acerca de los objetivos propuestos al inicio del trabajo con el fin de valorar si se han cumplido los objetivos o no.

8.1.1. Encontrar la forma de manejar y tratar la información que se encuentra en los logs.

El problema básico del que parte el proyecto es ser capaz de encontrar una forma de manejar toda la información recibida y dotarla de un sentido comprensible para el observador. Se considera que se ha encontrado una forma de manejar la información que se encontraba en Guernika, la utilización de la arquitectura de *Elastic Search* ha permitido tratar el grueso de la información que ha sido almacenada durante los dos cuatrimestres del año 2017. Se ha realizado una conversión de la información sin formato de los logs a una base de datos accesible que ha permitido recuperar y filtrar la información de forma refinada.

8.1.2. Utilizar la información contenida en los logs para realizar un estudio sobre lo ocurrido en Guernika.

El segundo punto que abarca este proyecto es realizar un estudio sobre lo que ha pasado en Guernika. Si bien la información almacenada durante un año en un servidor multiplataforma puede llegar a ser una cantidad de información abudante, se ha decidido centrar el centrar los esfuerzos y acotar el estudio de lo ocurrido en Guernika a través de las autenticaciones y la actividad que han realizado los usuarios durante el cuatrimestre. Utilizando la información refinada almacenada en Elastic Search se ha realizado con éxito un estudio sobre los ficheros auth.log que recogen tanto la autenticación de los usuarios como parte de sus actividades.

8.1.3. Utilizar el estudio para realizar un informe sobre el estado de la seguridad en Guernika.

El estudio que se ha realizado ha permitido realizar un informe de seguridad sobre Guernika a través del estudio de los ficheros *auth.log*, por lo que se considera que se ha cumplido con éxito el punto final de este proyecto que es extraer conclusiones sobre el estudio de sus logs.

8.1.1. Relación de asignaturas cursadas durante el grado que han permitido abordar este proyecto

Durante el grado se han cursado una serie de asignaturas que han permitido entender y afrontar este proyecto de una mejor forma que si no se dispusiera de tal formación, considero que las asignaturas que me han ayudado durante este proyecto han sido:

- **Criptografía y seguridad informática**: Debido a los métodos criptográficos vistos en clase.
- **Ingeniería de la seguridad y seguridad en dispositivos móviles**: Por la parte destinada al análisis de amenazas y posibles ataques además de la parte sobre protección de datos.
- **Sistemas operativos y arquitectura de computadores**: Debido al conocimiento y funcionamiento visto en clase sobre Linux.
- **Sistemas distribuidos**: Debido a la estructura cliente/servidor vista durante el curso así como de arquitecturas distribuidas.
- **Redes de ordenadores**: Debido al conocimiento sobre las conexiones entre computadores que se afronta en el proyecto.
- **Ingeniería del software**: Por la parte de identificación de requisitos.
- **Dirección de proyectos de desarrollo de software**: Debido a la parte de planificación y relación de requisitos.
- **Fundamentos de gestión empresarial**: Debido a la parte sobre el presupuesto y planificación.
- **Técnicas de búsqueda de la información y expresión oral y escrita**: Debido a que se enseña como citar y elaborar una estructura para el proyecto final de grado.

8.1.2. Personales

Este proyecto me ha permitido además aprender una herramienta nueva no vista durante el grado, que ha sido Elastic Search. Así como tener la oportunidad de manejar y aprender sobre una cantidad importante de datos de un servidor real y realizar un informe de seguridad sobre estos.

8.2.Problemas encontrados

El principal problema que se ha encontrado durante la realización del proyecto ha sido empezar a trabajar con una idea en concreto. Al tratarse de un proyecto sin una meta definida, ha sido tarea del alumno acotar y diferenciar cuál es la meta final. Quizá esto ha costado más trabajo que la implementación técnica así como las conclusiones finales, pues desde el principio surgían muchas ideas que no terminaban de cobrar forma, decidirse por una ha sido complicado y lo que ha ayudado a salvar este escollo ha sido un análisis más técnico del problema y sobretodo ser realista con la meta propuesta.

Por otro lado, otro de los problemas difíciles a los que se ha hecho frente es la limitación técnica del equipo sobre el que se implementó ELK. La potencia limitada del portátil ha hecho que los tiempos de análisis se alarguen más de lo previsto y cuando todos los logs se encontraban cargados en el portátil la mayoría de las veces se quedaba sin capacidad de procesamiento haciendo que el programa se cerrase. Esto dificultó muchísimo el análisis pues tenía que pensar mucho antes de realizar cualquier consulta, ya que una consulta que no reportara información útil y devolviera muchos registros suponía tumbar parte de la arquitectura. Se trata de un error de dimensionamiento que el tiempo disponible no permitió corregir.

Otro de los imprevistos fue que no se previó que los datos originales de los logs no iban a poder ser publicados debido a que interferían con las normativas de privacidad vigentes.

Por eso causó la necesidad de implementar el ofuscador de nombres de usuario.

Y finalmente por supuesto ha sido complicado manejar el tiempo para trabajar, y realizar el proyecto. Probablemente con el tiempo destinado al completo al proyecto se hubiera podido avanzar más lejos.

8.3.Líneas futuras

Tras la realización del proyecto se observan las siguientes líneas futuras como continuación de este trabajo:

1. **Mejora de la ofuscación de los datos**. En vez de realizar la ofuscación con la información ya almacenada en ElasticSearch se debería de plantear algún tipo de mecanismo que permita ofuscar la información de los usuarios antes de almacenarla. Para ello se propone modificar el plugin Grok de LogStash.
2. **Evaluar el informe realizado:** Evaluar el informe realizado con el fin de buscar algún tipo de fallo o comportamiento que no se esperaba por parte de los administradores del servidor.
3. **Mejorar el sistema de logs de Guernika:** Se propone utilizar ElasticSearch de forma activa, en vez de realizar una carga de logs pasados, disponer de ElasticSearch como monitor diario de logs y por tanto del estado actual de Guernika.
4. **Aplicar inteligencia artificial:** Este trabajo puede servir de inicio de otro tipo de trabajo que permita anticipar comportamientos o fallos pues se dispone de un análisis de al menos un curso lectivo que puede servir de entrenamiento para distintos modelos de inteligencia artificial.
5. **Seguir recolectado información para realizar informes anuales:** Si se sigue recolectando información de forma constante se puede comparar un periodo de tiempo con otro permitiendo detectar tendencias y patrones en los usuarios y lo que ocurre en el servidor.
6. **Análisis de las aplicaciones:** Realizar un informe sobre las aplicaciones que ya se encuentran en Guernika, no limitar el estudio a los logs relativos a ssh.
7. **Alerta a las conexiones remotas:** Muchas conexiones que llegan a Guernika pasan el filtro sobre el que está amparado el servidor, se debería de revisar la configuración del filtro por el que está amparado Guernika.

Chapter 9: English competitions

This document is about the study of a log collection from a Carlos III university server, named Guernika.

This server is in the computer lab of the security team and is used by the students and teachers to test applications and learn computer science. This server was created focusing on maintaining an environment the students can use to do their homework, this offers flexibility to the students because they can develop their applications even if they are out of the university.

In addition, this server is used by the teachers to evaluate the job done by the students. It is used as environment test because all the students, with their university account, can access to the computer by connecting via SSH. So in most of the subjects is used as the test environment to evaluate the job done by the students.

Using this server with multiple accounts requires a system to record all the activity done by every user. This method is called *logging.*

Logging is the technique used to store the information about what is happening in the server in files called *log*. A log file is just a raw file that contains information from certain programs.

There are two main problems with this type of files:

1. Dimensions and capacity: Storing all what is happening in a server can be a huge information, so the information is distributed in several files of the same type of log with the cost of capacity associated. Records from a long period of time can be a lot of files with Gigabytes of information.
2. There is not a standard way to write the information in a log file so the format of two different logs can be different.

This makes logs very difficult to read and handle for a human person.

In addition, the technique of logging is facing the rising of more cybersecurity attacks and threats. We can see often on the news or TV cybersecurity incidents like the attack to Sony Pictures, the spread of ramsonwares like Wannacry or Whiterrabbit, the break of the protocol WPA2 or the attack from the IoT botnet Mirai to the infrastructures of the United States.

So in this times, we need new techniques to evaluate the information the log files contain inside.

In this project, the student received a log collection from the server Guernika to make a study of what happened in the server using the log files.

So the first thing to do in this project is to understand what type of server is Guernika.

Guernika is a server that runs Linux. To be more specific, this computer runs Debian, an operating system based on the Linux kernel with GNU tools. The Debian version used is Debian 9. All the students can connect to the server via SSH, so the server is internet accessible and no VPN is needed.

Keeping in mind that Guernika is a Linux multiuser server and has internet access, the next step is to study what type of log the student has received.

The log collection received contains a directory named */var/log*. In this folder there are the following logs:

- 13 *alternatives.log*
- 356 *auth.log*
- 356 *daemon.log*
- 6 *debug.log*
- 14 *dpkg.log*
- 198 *kern.log*
- 224 *mail.log*

There is one log of every type, the rest all compressed in .gz format.

And the following directories:

- *apt*: contains 13 *history.log* and 13 *term.log.*
- *firebird*: empty folder.
- *fsck*: contains two files checkfs and checkroot.
- *gdm3*: empty folder.
- *installer*: contains the subdirectory cdebconf and inside there are two files *questions.conf* and *templates.dat* finally in the root directory there the following files: *installer hardware-summary, lsb-release, partman, status and syslog*
- *lightdm*: contains *lightdm.log* and *lightdm.log.old*, *x-0.log,x-0.log.old*, *x-0-greeter.log* and *x-0-greeter.log.old*
- *munin*: contains 9 *munin-node.log*
- *ntpstats*: empty folder.
- *Samba*: empty folder.
- *speech-dispacher*: empty folder.

So a quick analysis of the files received reveals that is a copy of the directory */var/log*, this directory is used by Linux distributions to store by convention all the critical logs recorded by the operating system.

The next step is focusing on the project because there are too many files to study. There are log files related to the use of apt, a package manager to install new programs, there are log files related to the running kernel and programs like *daemon.log* and *kern.log*. There is also information about the authentication of users inside Guernika in logs like *auth.log*, so we can observe here the following parts to study:

1. The running programs inside Guernika.
2. The users of Guernika and their behavior.
3. The exposure of Guernika to the internet.

The student decided to use the *auth.log* to study the parts 2 and 3, because this is considered the most complete log, that type of log store the information about ssh connections and what is executed by every user.

The next step was to find the way to analyze all the *auth.log* files. To do this job the student decided to analyze the actual software that is used in the market to analysis this type of files, in the case that there is not an efficient way to analyze this files the student decided to make new tools to do this job, but this is the last option because we do not want to create a new way to analyze the log files we want to focus on the process to analyze and learn about what log file contains.

In the study done about the market, we found that there are a lot of tools used by this type of analysis, many of them are closed software. We want to avoid closed and commercial software because we want to use tools we can configure and adapt to our situation. In addition, we do not want to expend too much money in closed software because is a research study and we want that this job can be replicated by everyone. Some of this tools are free software GreyLog and ElasticStack, a quick research indicates that GreyLog is based on ElasticStack and we found that ElasticStack can be used to analyze our files, so we decided to go to the original source of software and use ElasticStack.

ElasticStack is a distributed piece of software that contains the following parts:

1. ElasticSearch: Is the search engine that stores the information contained in the log files.
2. FileBeat: This part is configured to read a directory a send the log files with a registry to LogStash to be processed.
3. LogStash: Is the part used to evaluate the log files and chop the information in small pieces. Is the parser.
4. Kibana: This part is used to visualize with the web browser the information stored in ElasticsSearch.

The next step is to make a design about how we can use ElasticStack to solve the problem of parse all the log files and later recover the information inside ElasticSearch.

There were 3 types of design: One is based on running all the components in one machine, the next is based on the idea of using different machines to execute different parts of Elastic Stack and the last one is based on the idea of using a cloud machine like Amazon AWS or Windows Azure to execute all the Elastic Stack.

The first one was used as the final design because we need software that can process all the previous records stored in the log files so the process is needed to be executed only once. Keeping the idea of one execution we can discard the execution on a cloud service or on a distributed computer system because we do not need elastic resources or scalability.

The final design is based on the following guidelines:

FileBeat is used to read the path were the LogFiles are stored and send the read log files to LogStash. FileBeat is configured to run in the ip localhost and port 5044 and send the information to LogStash in the ip localhost and port 5000. LogStash is configured to receive the information from FileBeat and process the auth.log files. Finally, LogStash sends the refined information to ElasticSearch in the ip localhost and port 9400. This part of the design is the part configured to load and process all the logs in to the architecture.

To recover the information stored in the architecture we will use Kibana in the IP localhost and port 5601. This part is used to analyze all the data stored in Elasticsearch.

The next step is configuring the design in our architecture. First, we need to install all software pieces in the computer used to execute Elastictack.

For that porpoise, we need to download the software from Elasticsearch project. We used Debian packages called .deb, once we have the software downloaded in the computer we can install the software using the package manager dpkg.

The final step to work the architecture is configuring the services described before. We need to edit the configuration of FileBeat to recover the log files. This can be done by editing the file */etc/share/filebeatPROCESS_AUTH_ALL.yml*. We need to configure the path where are the log files are stored and where Filebeat must redirect the log files, this place will be LogStash in IP localhost and port 5000.

Now we need to edit the configuration of LogStash and ElasticSearch, first we need to edit the file in the path */etc/filebeat/pipeline_custom.conf*. This file is critical because we need to specify how LogStash will process the log raw file. In this file we need to specify that we want the following data:

- Timestamp: because we want to filter all the events by date.
- Usernames: to trace the activity by the user
- IP
- Authentication state: to know what happened with the authentication.
- Authentication method: to know what methods are used in the process of authentication.
- Commands used: to trace the activity.
- The use of sudo and root: to trace the activity of the root user.

- Path used in the command: To know part of the intention of the user
- Geocalization: To trace where the connections are done.

Finally, we need to configure Logstash to send all the refined data to Elasticsearch in the IP localhost and port 9400.

The next step is configuring Elasticsearch to listen in the IP localhost and port 9400.

We have done all the configurations needed, so now we can execute all the pieces of software in this order:

First, we need to deploy Elasticsearch with the command:

systemctl start elasticsearch

The next step is deploy Logstash with the command:

/usr/share/logstash/bin/logstash -f /etc/filebeat/pipeline_custom.conf --config.reload.automatic --path.settings=/etc/logstash

With ElasticSearch and Logstash running now we can start Filebeat:

/usr/share/filebeat/bin/filebeat -e -c /etc/filebeat/filebeatPROCESS_AUTH_ALL.yml -d "publish"

This pipeline process will start to read all the Log files, process the raw files and store the refined data in Elasticsearch. This process is called load process.

This process was a long process because there were a lot of log files to store and lasted several hours.

Once the process has ended, we can stop Logstash and Filebeat because in this process we need only one load process to store all the old log files.

Now we got all the refined data stored in the search engine Elasticsearch. This point is the start of the next phase, the analysis of the refined information stored. To do this we need to use Kibana.

Kibana is used to read all the data store in Elasticsearch using a web browser, so first we need to start the service of Kibana:

systemctl start kibana

Now we can connect using our favorite web browser to the url *localhost:5601* if all was done nicely we can see the web site offered by Kibana, the front page of the service.

By default, Kibana is not configured to read any data so the first thing we need to do is specify what data Kibana should read. We can do that in the part of Kibana called management.

In this part, we can see the Elasticsearch indexes. An index is the logical way of how Elasticsearch stores the data.

So we need to specify what index Kibana will use. In this case, we want to use all the stored records from the old log files so we can use an index pattern to specify all the files.

The pattern is *logstash** this pattern specifies that we want to math all the indexes that contain in their name the string logstash and the ones starting with it.

Now that we have all the data configured, we can start using Kibana to visualize the data stored.

We can test what we want to search with the tool discover. This tool is used to search for data inside the previously defined Kibana patterns. Is based on the Lucene syntax and we can save our searches with a name to use it later.

But featured key we want to use is the tool visualize. With this tool we can use our previous searches tested with the tool discover and use it to make fancy graphics.

Kibana has a lot of types of graphics and charts to report the information inside the indexes, we can choose graphics like horizontal toolbar, vertical toolbar, pie chart, lists, metrics, maps and more.

So now start to make out graphics to know what happened. To do this we used the following charts:

- Programs that have written in the log files.
- Authentications methods.
- State of the authentications.
- State of the authentications by day.
- State of the authentication filtered by the state of authentication accepted.
- State of the authentication filtered by the state of authentication failed.
- State of the authentication filtered by the state of authentication invalid.
- State of the authentication filtered by the state of authentication disconnecting.
- State of the authentication filtered by the state of authentication disconnecting to many connections.
- Geolocation by continent.
- Geolocation by country.
- Geolocation by city.
- Commands used by the user root.
- Commands that have been tried to be executed as root.
- Users that have succeeded in executing commands as root.
- Users that tried to execute commands as root.
- Root loggings.
- Admin loggings.
- Guest loggings.

With this information, we can make a report of what happened in the server in the period studied by records inside the log files.

This study goes from 2017-02-01 to 2017-11-06.

We can see that the top 5 programs who have written information in the auth log files are snoopy, cron, sshd, system-logind and sytemd. Snoopy is used to track user´s activity, cron to schedule jobs, sshd is the daemon used in ssh connections, system-logind and system are daemon to use services in the operating system.

About the authentication methods, we can see that the 98% of the authentications are done with the combination of username and password and the 1.59% are done with publickey.

The state of authentication chart reports that the 93% of the authentications succeeded with the state accepted, on the other hand 5.29% are failed and 1.38% invalid. The main difference between invalid and failed is that failed is when the credentials are incorrect like username or password and invalid is when someone is trying to use a user that is not allowed to log in but is registered in the system.

Using in details the chart *State of the authentications* we can see what happened in the whole time studied. The time studied is a complete course of university so we can see dates with too many connections as one key date is going to happen in time like the correction of a practice or exams, this is a normal and expected behavior but we can see in the month of April that has a lack of connections from high connections goes to zero.

This can be interpreted as two things: Something happened in April like a disconnection or this can be a failure recording.

Using the charts:

State of the authentications by day, State of the authentication filtered by the state of authentication accepted,State of the authentication filtered by the state of authentication failed. State of the authentication filtered by the state of authentication invalid. State of the authentication filtered by the state of authentication disconnecting. State of the authentication filtered by the state of authentication disconnecting to many connections.

We can see the previously observed behavior in the Guernika in the month of April, we can see the lack of connections in all the authentication states individually-

On the other hand, we studied the geolocation of the connections done to Guernika. We can see that the top country with more connections is Spain. This is a normal behavior because the University of Carlos III is Leganés, Madrid. But we can see connections from outside Spain. We expected to have connections from Asian countries but we got only a few from Iran. This is an abnormal behavior because we expected connections from China and Russia countries that have a great impact on the internet.

So we can assume that Guernika is under a firewall or a security system that is blocking connections from certain countries.

As expected we can see that the top cities are Madrid and Leganes because are the nearby cities and of course the top continent where are the connections from is Europe.

Studying the use of root and using the following charts:

Commands used by the user root, commands that have been tried to be executed as root, users that have succeeded executing commands as root, users that tried to execute commands as root. We can see anomalies there are users trying to execute commands as root that doesn't have enough privileges to do that, as we can see in the records there are not coordinated attacks to do harmful actions on the system with a few exceptions. But we can't considered this as malicious attacks because most of the users are students testing what can do and what can´t.

And using the last charts: *Root loggings, Admin loggings, Guest loggings, w*e can observe brute force attacks maybe by spam bots or automated crawlers trying to log in the machine by brute force attacks. We observe this because all the successful loggings of the user root are done using the public key but on the other hand all the failed loggings are done with username and password in addition all the failed loggings are done from outside of Spain.

If we use other usernames to do this study like root, admin and guest for example, we can observe the same behavior so we can conclude that Guernika is under brute force attacks trying to log in in the computer. But there are not great attacks just automated check in to discover the services in Guernika.

So in conclusion we made a final report:

The report on the auth.log files allows us to conclude that the status of the studied SSH connections does not report the evidence of a serious attack on Guernika.

However, it is recommended to have the following considerations:

1. Consider auditing the commands and activity of the users complementing the existence of Snoopy with the Red Hat tool available for Linux distributions audited.
2. Consider disabling the root user through direct access to SSH, is found to enable root access through another intermediate user.
3. Review the accesses by the public key both at the root and the rest of the users.
4. Review the activity of the root user.
5. Take into consideration and review the period of inactivity of the server during the month of April.
6. Review the permissions of the folders home users.
7. Review the permissions of some important binaries, which are not accessible with public permits.
8. Take into account the remote access by bots and consider drawing the received IPS with blacklists.
9. Consider whether to use the use of rkhunter in order to search for rootkits and possible threats and contrast with the commands made during the session.

The planning done to do this study is the following one:

Analysis

Duration: 16 days.

This is the phase that involves both the analysis of the problem and the analysis and study of the tools used in the project.

- Analysis of the problem.
- Analysis of possible solutions.
- Search for alternatives.
- Study of the tools involved.
- Architecture design to solve the problem.
- Design of alternative architectures.

Preparation of the environment

Duration: 4 days.

This is the phase in which the design is applied and the tools are prepared for its use.

- Installation of the operating system.
- Installation of the necessary software dependencies for the record manager.
- Installation of the Logs manager.
- Adjust and ready Logs handler.

Test and validation of tools

Duration 1 day.

This is the phase in which it is validated and verifies that the installed environment corresponds to the architectural design carried out.

Loading data

Duration: 4 days.

This is the phase in which the Logs are loaded to the platform and validates that the quality of the data loaded on the platform is correct.

- Load of the Logs.
- Verification of the data load.

Conclusions

Duration: 12 days.

This is the phase in which the data obtained is studied and analyzed.

- Identification of users.
- Geolocation of users.
- Behavior monitoring during the session.

- Traceability of the commands used.
- Checking attack attempts.
- User classification.
- Discarding of false positives.

Documentation and obfuscation of data

This is the phase in which the detailed study of the study carried out is recorded.

- Drafting of the project's memory.
- Last conclusions.

Conclusions

The next approximations of the proposed objectives are considered at the beginning of the work in order to assess whether the objectives have been met or not.

Find the way to handle and treat the information found in the records.

The basic problem from which the project starts is to be able to find a way to handle all the information received and to provide it with a meaning that is understandable to the observer. A search has been made of a tool that can be found in Guernika, the use of the Elastic Search architecture has allowed us to deal with the bulk of the information stored during the semesters of the year 2017. A conversion of the information to the format of the records to an accessible database that has allowed to recover and filter the information in a refined way.

Use the information contained in the registers to carry out a study about what happened in Guernika.

The second point that this project covers is a study of what happened in Guernika. Although the information stored during a year in a multiplatform server can reach a quantity of complete information, it has been decided to focus the efforts and undertake the study of what happened in Guernika through the authentications and the activity that the users have carried out during the semester. Using the refined information stored in Elastic Search, a study has been carried out on the files that recognize both the authentication of the users and part of their activities.

Use the study to make a report on the state of security in Guernika.

The study that has been conducted has allowed a security report on Guernika through the study of auth.log archives, so it is considered that the final point of this project has been successfully completed, which is to draw conclusions about the study of the logs.

Bibliografía

[1] ptsecurity.com, "ptsecurity," 17 7 2017. [En línea]. Disponible: https://www.ptsecurity.com/ww-en/about/news/293941/. [Último acceso 16 9 2018].

[2] S. R. E. Yves Younan, «welivesecurity,» Sourcefire, [En línea]. Disponible: https://courses.cs.washington.edu/courses/cse484/14au/reading/25-years-vulnerabilities.pdf. [Último acceso: 16 9 2018].

[3] T. M. Corporation, "cve," 16 9 2018. [En línea]. Disponible: https://cve.mitre.org/. [Último acceso 16 9 2018].

[4] I. T. Laboratory, "nvd.nist.gov," 16 9 2018. [En línea]. Disponible: https://nvd.nist.gov/general/nvd-dashboard. [Último acceso 16 9 2018].

[5] EFE, "efe.com," 11 1 2018. [En línea]. Disponible: - https://www.efe.com/efe/espana/sociedad/espana-registra-120-000-incidentes-en-ciberseguridad-2017-cifra-record/10004-3489190. [Último acceso 16 9 2018].

[6] G. U. o. Technology, "meltdownattack," 2018. [En línea]. Disponible: https://meltdownattack.com/. [Último acceso 16 9 2018].

[7] Avast, "avast.com," 2018. [En línea]. Disponible: https://www.avast.com/es-es/c-ransomware. [Último acceso 16 9 2018].

[8] Xataka, 7 6 2018. [En línea]. Disponible: https://www.xataka.com/basics/como-saber-si-una-web-esta-minando-bitcoins-aprovechandose-de-tu-visita-y-como-evitarlo. [Último acceso 16 9 2018].

[9] O. d. s. d. internatua, "osi.es," 2018. [En línea]. Disponible: https://www.osi.es/es/servicio-antibotnet/info/mirai. [Último acceso 16 9 2018].

[10] e. Rosa Jiménez Cano, "elpais.es," 25 11 2014. [En línea]. Disponible: https://elpais.com/tecnologia/2014/11/25/actualidad/1416904284_635758.html. [Último acceso 16 9 2018].

[11] J. Sánchez, "elmundo.es," 16 2 2017. [En línea]. Disponible: https://www.abc.es/tecnologia/redes/abci-yahoo-asume-brecha-seguridad-debe-falsificacion-cookies-entre-usuarios-201702161136_noticia.html. [Último acceso 16 9 2018].

[12] I. Ramirez, "xatakandroid.com," 14 12 2017. [En línea]. Disponible: https://www.xatakandroid.com/seguridad/comprueba-si-tu-android-es-vulnerable-al-ataque-blueborne-con-esta-aplicacion. [Último acceso 16 9 2018].

[13] M. Vanhoef, imec-DistriNet, 16 10 2017. [En línea]. Disponible: https://www.krackattacks.com/. [Último acceso 16 9 2018].

[14] Wikipedia, "es.wikipedia.org," 18 6 2018. [En línea]. Disponible: https://es.wikipedia.org/wiki/Log_(inform%C3%A1tica). [Último acceso 16 9 2018].

[15] J. d. g. Español, 17 6 1997. [En línea]. Disponible: https://bit.ly/2xeYPnJ. [Último acceso 16 9 2018].

[16] C. Gestora, 20 6 1994. [En línea]. Disponible: https://bit.ly/2NftRpR. [Último acceso 16 9 2018].

[17] "Normativa de utilización aulas informáticas," 3 3 1993. [En línea]. Disponible: https://bit.ly/2QyBhCi. [Último acceso 16 9 2018].

[18] "sumologic.com," [En línea]. Disponible: https://www.sumologic.com/security/platform-security/. [Último acceso 13 8 2018].

[19] SolarWings, "papertrailapp.com," [En línea]. Disponible: https://papertrailapp.com/plans/large. [Último acceso 13 8 2018].

[20] E. Search, "elastic.co," [En línea]. Disponible: https://www.elastic.co/downloads/elasticsearch. [Último acceso 22 9 2018].

[21] E. Search, "elastic.co," [En línea]. Disponible: https://www.elastic.co/downloads/logstash. [Último acceso 22 9 2018].

[22] E. Search, "elastic.co," [En línea]. Disponible: https://www.elastic.co/downloads/beats. [Último acceso 22 9 2018].

[23] E. Search, "elastic.co," [En línea]. Disponible: https://www.elastic.co/downloads/kibana. [Último acceso 22 9 2018].

[24] E. BV, "elastic.co," Elasticsearch BV, 2018. [En línea]. Disponible: https://www.elastic.co/guide/en/kibana/6.3/index.html. [Último acceso 20 9 2018].

[25] M. Collins, Network Security Through Data Analysis, 2nd Edition, O'Reilly Media, September 2017.

[26] S. M. A. a. P. K. L. T. Symul, "Applied Physics Letters," 7 6 2011. [En línea]. Disponible: https://aip.scitation.org/doi/10.1063/1.3597793. [Último acceso 20 9 2018].

[27] S. A. A. L. N. N. V. S. P. L. a. T. S. J. Y. Haw, 11 5 2015. [En línea]. Disponible: https://journals.aps.org/prapplied/abstract/10.1103/PhysRevApplied.3.054004. [Último acceso 20 9 2018].

[28] ANU, "qrng.anu.edu.au," 11 5 2015. [En línea]. Disponible: https://qrng.anu.edu.au. [Último acceso 20 9 2018].

[29] M. A. Eriksen, "github," 3 7 2015. [En línea]. Disponible: https://github.com/a2o/snoopy. [Último acceso 18 9 2018].

[30] M. A. Eriksen, "github," [En línea]. Disponible: https://github.com/a2o/snoopy/blob/master/doc/FAQ.md#5-i-see-no-snoopy-output-after-initial-user-login. [Último acceso 18 9 2018].

[31] "http://www.lab.inf.uc3m.es," uc3m, [En línea]. Disponible: http://www.lab.inf.uc3m.es/wp-content/docs/manual_conexionSSH_V1.0.pdf. [Último acceso 18 9 2018].

[32] H. Mederos, "laopinioncoruna.es," 21 7 2017. [En línea]. Disponible: https://www.laopinioncoruna.es/contraportada/2017/07/21/rusia-china-son-estados-lanzan/1201936.html. [Último acceso 18 9 2018].

[33] I. MXToolBox. [En línea]. Disponible: https://mxtoolbox.com/blacklists.aspx. [Último acceso 18 9 2018].

[34] RedHat, "redhat.com," [En línea]. Disponible: https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/6/html/security_guide/chap-system_auditing. [Último acceso 22 9 2018].

[35] ElasticSearch. [En línea]. Disponible: https://www.elastic.co/guide/en/logstash/current/logstash-config-for-filebeat-modules.html. [Último acceso 20 9 2018].

[36] A. foundation, "apache.org," 1 2004. [En línea]. Disponible: https://www.apache.org/licenses/LICENSE-2.0. [Último acceso 24 9 2018].

[37] Debian.org, "Debian.org," 27 8 2018. [En línea]. Disponible: https://www.debian.org/legal/licenses/. [Último acceso 24 9 2018].

Anexos

Configuración de LogStash [35]

```
input {
  beats {
    port => 5044
    host => "localhost"
  }
}
filter {
     grok {
       match => { "message" => ["%{SYSLOGTIMESTAMP:[system][auth][timestamp]}
%{SYSLOGHOST:[system][auth][hostname]} sshd(?:\[%{POSINT:[system][auth][pid]}\])?:
%{DATA:[system][auth][ssh][event]} %{DATA:[system][auth][ssh][method]} for (invalid user
)?%{DATA:[system][auth][user]} from %{IPORHOST:[system][auth][ssh][ip]} port
%{NUMBER:[system][auth][ssh][port]} ssh2(:
%{GREEDYDATA:[system][auth][ssh][signature]})?",

                "%{SYSLOGTIMESTAMP:[system][auth][timestamp]}
%{SYSLOGHOST:[system][auth][hostname]} sshd(?:\[%{POSINT:[system][auth][pid]}\])?:
%{DATA:[system][auth][ssh][event]} user %{DATA:[system][auth][user]} from
%{IPORHOST:[system][auth][ssh][ip]}",

                "%{SYSLOGTIMESTAMP:[system][auth][timestamp]}
%{SYSLOGHOST:[system][auth][hostname]} sshd(?:\[%{POSINT:[system][auth][pid]}\])?:
Did not receive identification string from %{IPORHOST:[system][auth][ssh][dropped_ip]}",

                "%{SYSLOGTIMESTAMP:[system][auth][timestamp]}
%{SYSLOGHOST:[system][auth][hostname]} sudo(?:\[%{POSINT:[system][auth][pid]}\])?:
\s*%{DATA:[system][auth][user]} :( %{DATA:[system][auth][sudo][error]} ;)?
TTY=%{DATA:[system][auth][sudo][tty]} ; PWD=%{DATA:[system][auth][sudo][pwd]} ;
USER=%{DATA:[system][auth][sudo][user]} ;
COMMAND=%{GREEDYDATA:[system][auth][sudo][command]}",

                "%{SYSLOGTIMESTAMP:[system][auth][timestamp]}
%{SYSLOGHOST:[system][auth][hostname]}
groupadd(?:\[%{POSINT:[system][auth][pid]}\])?: new group:
name=%{DATA:system.auth.groupadd.name}, GID=%{NUMBER:system.auth.groupadd.gid}",
```

```
        "%{SYSLOGTIMESTAMP:[system][auth][timestamp]}
%{SYSLOGHOST:[system][auth][hostname]}
useradd(?:\[%{POSINT:[system][auth][pid]}\])?: new user:
name=%{DATA:[system][auth][user][add][name]},
UID=%{NUMBER:[system][auth][user][add][uid]},
GID=%{NUMBER:[system][auth][user][add][gid]},
home=%{DATA:[system][auth][user][add][home]},
shell=%{DATA:[system][auth][user][add][shell]}$",

        "%{SYSLOGTIMESTAMP:[system][auth][timestamp]}
%{SYSLOGHOST:[system][auth][hostname]}
%{DATA:[system][auth][program]}(?:\[%{POSINT:[system][auth][pid]}\])?:
%{GREEDYMULTILINE:[system][auth][message]}"] }

    pattern_definitions => {

      "GREEDYMULTILINE"=> "(.|\n)*"

    }

    remove_field => "message"

  }

  date {

    match => [ "[system][auth][timestamp]", "MMM  d HH:mm:ss", "MMM dd HH:mm:ss" ]

  }

  geoip {

    source => "src_ip"

    target => "geoip"

    add_field => [ "[geoip][coordinates]", "%{[geoip][longitude]}" ]

    add_field => [ "[geoip][coordinates]", "%{[geoip][latitude]}"  ]

  }

  mutate {

    convert => [ "[geoip][coordinates]", "float"]

  }

}

output {

  elasticsearch {
```

index => "logauth-%{+YYYY.MM.dd}"

hosts => ["localhost:9200"]

}

stdout { codec => rubydebug }

}

Código Python del ofuscador

```
import requests
import json
import csv
import onetimepad
import sys

# class used to store temporally the real username+ciphered+key
class UsernameSwap:
    real_username = ""
    fake_username = ""
    key_xor = ""
    def __init__(self,real,fake,key):
        self.real_username = real
        self.fake_username = fake
        self.key_xor = key

# call to the ANU api
def api_call(to_call):
    url = "https://qrng.anu.edu.au/API/jsonI.php"
    if to_call is None:
        param_len = "1"
    else:
        param_len = str(to_call)
    param_size = "6"
    param_fixed_type_h16 = "hex16"

    string_i = "?"
    string_a = "&"
    string_len = "length="
    string_typ = "type="
    string_siz = "size="

    url = url + string_i + string_len + param_len + string_a + string_typ +
param_fixed_type_h16 + string_a + string_siz + param_size
    print(url)
    response = requests.get(url)

    if response.status_code == 200:
        content = json.loads(response.content)
        return content['data']

# read the userfile
def read_file(path_file):
    if path_file is not None:
        usernames = []
        with open(path_file) as csvf:
            reader = csv.reader(csvf,delimiter=",")
            for row in reader:
```

```
                usernames.append(row[0])
        return usernames
    else:
        return -1



# swap the real username with the ciphered
def swap_usernames(usernames):
    if usernames is not None:
        usernames_swapped = []
        usernames_alt = api_call(len(usernames))
        i = 0
        for user in usernames:
            aux_key = usernames_alt[i][:len(user)]
            otp = onetimepad.encrypt(user,aux_key)
            aux_user = UsernameSwap(user, otp, aux_key)
            usernames_swapped.append(aux_user)
            i+=1
        return usernames_swapped
    else:
        return -1

# write a backup file with username+ciphered username+key
def write_backup_file(usersnames_swapped):
    backup = open("usernames_swapped.txt","w")
    for user in usersnames_swapped:
        backup.write(user.real_username+"="+user.fake_username+"
key:"+user.key_xor+"\n")
    backup.close()

# print the ciphered usernames
def output_swapped(usernames_swapped):
    for user in usernames_swapped:
        print(user.fake_username)

# workflow
def process():
    usernames_file = read_file("users.csv")
    usernames_swapped = swap_usernames(usernames_file)
    write_backup_file(usernames_swapped)
    output_swapped(usernames_swapped)

# MAIN
process()
```

Requirements.txt del ofuscador

El fichero requirements.txt se trata de un fichero que contiene las dependencias de las librerías utilizadas en el programa. En este caso se trata de las librerías requests,json, csv, onetimepad y sys

- requests
- json
- csv
- onetimepad
- sys

Para instalar las dependecias bastaría con utilizar el gestor de dependencias nativo de Python, pip.

pip install requirements.txt

Glosario de comandos

cd: Change directory, sirve para cambiar el directorio actual.

curl: Comando utilizado para enviar o recibir información a un servidor mediante distintos protocolos.

Dpkg: gestor de paquetes principal de Debian. Permite instalar y modificar paquetes en un sistema.

Grep: Comando utilizado para realizar búsquedas de texto plano en ficheros mediante expresiones regulares.

Gunzip: Comando utilizado para descomprimir o comprimir ficheros en distintos formatos

Kill: Sirve para parar o finalizar un proceso que se encuentra ejecutando en el sistema operativo.

Ls: Comando utilizado para listar los ficheros y directorios que se encuentran en un directorio.

Ps: Process status, muestra los procesos que se están ejecutando en un sistema.

Systemctl: Comando de consola que sirve para controlar systemd, que se encarga de manejar los servicios del sistema.

Vim: editor de línea de comandos flexible.

/usr/share/logstash/bin/logstash: Comando que contiene la ruta absoluta que se utiliza para desplegar logstash.

/usr/share/filebeat/bin/filebeat: Comando que contiene la ruta absoluta que se utiliza para desplegar filebeat.

Licencias software utilizadas

ElasticSearch, utiliza una licencia de software libre basada en la licencia Apache 2.0. [36]

Debian, licencia de software libre del proyecto Debian. [37]

Listado completo de ciudades

Madrid	**36,966**
Leganés	4,698
Barcelona	2,083
Getafe	1,741
Mostoles	1,671
Oleiros	1,585
Fuenlabrada	1,177
Alcobendas	1,049
Toledo	980
Castelló de la Plana	967
Alcorcón	841
Parla	833
Las Rozas de Madrid	598
Madridejos	585
Majadahonda	553
Quintanar de la Orden	499
Cordova	444
Maliano	423
Los Navalucillos	408
Fortuna	389
Nambroca	361
Murcia	338
Alcalá de Henares	301
Granada	300
Yebes	284
Valdemoro	268
Teià	250
Torre-Cardela	229
Colmenar Viejo	228
Pozuelo de Alarcón	224
Fuente el Saz	220
San Sebastián de los Reyes	219
Rivas-Vaciamadrid	206
Torrejón de Ardoz	183

Ashburn	181
Valdeolmos	171
Mountain View	167
Collado Villalba	161
Torrelavega	158
Valencia	153
Torrejon de la Calzada	149
Talavera de la Reina	147
Alhaurin el Grande	141
Magan	141
Corral de Almaguer	140
Esquivias	139
La Pueblanueva	139
El Espartal	137
Algemesi	134
Illescas	124
Yuncler	123
Los Yebenes	120
San Fernando de Henares	116
Menasalbas	113
Villaviciosa de Odon	109
Ciudad Real	106
Arroyomolinos	102
Fuengirola	102
Morata de Tajuna	97
Aranjuez	94
Ávila	87
Villacanas	86
El Escorial	85
Vila	83
Carranque	78
Ciempozuelos	74
Torrijos	72
Cobeja	67
Montefrio	59
Pinto	59
Benaocaz	55
Ronda	55
Becerril de la Sierra	54
Olbendorf	53
Navalcarnero	52
Alicante	50

Arcos de la Frontera	50
Coslada	50
Marbella	50
Jaén	48
A Coruña	47
Manzanares el Real	46
Málaga	46
Bochum	44
New York	43
Villanueva de la Serena	41
Yuncos	40
Zaragoza	40
Guadalajara	38
Sant Sadurni d'Anoia	37
Boadilla del Monte	36
Valles	36
Villaluenga	35
Arlington Heights	33
Pepino	33
Canedo	32
Majorca	30
Alhaurin de la Torre	29
Ocana	29
Vélez-Málaga	29
Algete	28
Cazalegas	28
Mocejon	28
Molina de Segura	28
Cedillo del Condado	27
El Viso de San Juan	27
Albaida	24
Bilbao	24
Grinon	24
Soria	24
Arucas	23
Vitoria-Gasteiz	23
Beniajan	22
Guadarrama	22
Las Gabias	22
Valladolid	22
Hoyo de Manzanares	21
Bucharest	20
Mentrida	20

Ibiza Town	19
Getxo	18
La Puebla de Montalban	18
Segovia	18
Camarenilla	17
Cistierna	17
Salobreña	17
Tres Cantos	17
Hoest	16
Yunquera	16
Tarazona de la Mancha	15
Valdemorillo	15
Dallas	14
Nuevo Baztan	14
Ontigola	14
Torres de la Alameda	14
Villanueva de la Canada	14
Arganda	13
Mutxamel	13
León	12
Moraleja de Enmedio	12
Oruro	12
Sant Joan	12
Abidjan	11
Boardman	11
Casasimarro	11
Flores	10
La Plata	10
Lucena	10
Mejorada del Campo	10
Miranda de Ebro	10
San Antonio	10
Tordesillas	10
Viña del Mar	10
Almería	9
Budakeszi	9
Humanes de Madrid	9
Mandeville	9
Oviedo	9
Pamplona	9
Sabadell	9
San Javier	9
Timișoara	9

Villanueva de Alcardete	9
Villasequilla de Yepes	9
Anover de Tajo	8
L'Hospitalet de Llobregat	8
Valmojado	8
Albacete	7
Amsterdam	7
Arroyo de la Miel	7
Burguillos de Toledo	7
Chiloeches	7
Villarrobledo	7
Las Ventas de Retamosa	6
Los Angeles	6
Reus	6
Seville	6
Vilagarcia de Arousa	6
Villalbilla	6
Cuellar	5
Dos Hermanas	5
El Burgo de Osma	5
Madronera	5
Meco	5
Olias del Rey	5
Sant Cugat del Vallès	5
Snina	5
Zuera	5
Almajalejo	4
Aveiro	4
Collbato	4
Copenhagen	4
Galaroza	4
Helsinki	4
Icod de los Vinos	4
La Paz	4
Montreal	4
Naples	4
Parets del Vallès	4
Roquetas de Mar	4
Salamanca	4
Vilassar de Mar	4
Washington	4
Azuqueca de Henares	3
Brenes	3

Cajar	3
Chisinau	3
Dublin	3
Galapagar	3
Igualada	3
Mungia	3
Bubion	2
Calvià	2
Camarma de Esteruelas	2
Campo Real	2
Cartagena	2
Casarrubios del Monte	2
Casavieja	2
Cáceres	2
Escalona	2
Frankfurt am Main	2
Fremont	2
Gijón	2
Hampstead	2
Mexico City	2
Rus	2
Serranillos del Valle	2
Torrelodones	2
Vilanova i la Geltrú	2
Alcalá de Guadaira	1
Arcenillas	1
Athens	1
Avilés	1
Brooklyn	1
Cadalso de los Vidrios	1
Camargo	1
Jacksonville	1
Los Molares	1
San Agustin del Guadalix	1
Sesena	1
Talamanca	1
Valdepenas	1
Vienna	1

Tabla 53- Listado completo de ciudades con conexiones realizadas a Guernika